Guidelines regarding submission of your assignment:

1. First page should only contain Name, Roll Number, Email ID and Index of questions with serial number and page numbers should be written. Every separate problem should start on separate page. The Hard Copy is to be submitted by 27 Sept 2011 by 5.00 P.M.
2. Input should be taken from a text file in which if the numbers are more than one than the numbers are separated by commas. If input consists of multiple sets than each set is on separate line.
3. Output should be generated in a text file. Output should also include the time taken to run the program and Memory size used by the program.
4. In a program containing arithmetic's it should give number of multiplications and number of additions/subtractions. In searching it should also give number of comparisons and in sorting it should give number of comparisons as well as number of swaps. In case of data structures it should give number of accesses, number of shifts etc.
5. Partial marks will be given for correct answer which is presented in a nice and structured way and analysed properly. Also give proof of correctness of your answer wherever possible. Answer having most efficient answer will fetch more marks.
6. Assignments submitted after due time will be marked but will not be considered for the purpose of grading.
7. Any unfair practice of copying may lead to cancellation of current and previous assignments also.
8. Soft Copy is to be submitted in three phases First five questions till 30 August,2011 5.00 P.M., Next Five Questions 10th September,2011 5.00 P.M. and Next Six Questions 27 September,2011 5.00 P.M.
9. All the Roll numbers ending with 1 should do 1st question of every section, all the roll numbers ending with 2 should do 2nd question of every section and so on.
10. The intention of giving a specific input file for all programs does not mean that your program should work only for this input. While evaluating other inputs will also be tested and these should generate correct output, So your program should be a general program for all inputs of the given kind in any question.
11. File name should be r<rollno>_<sectionnumber>_questionnumber>.cpp for example r100903078_1_1.cpp
12. Input file should be in1_1.txt and output file should be out_100903078_sec2_4.txt
13. Students should only submit program file and output file

Section 1:

Factorial Programs

1. Write an efficient program to find out the individual factorial of a set of n numbers given in the list. First number is the value of n.
Input File
7
5
12
20
37
49
53
69
Output File
5=120
12=479001600
20=2432902008176640000
37=13763753091226345046315979581580902400000000
49=608281864034267560872252163321295376887552831379210240000000000
53=4274883284060025564298013753389399649690343788366813724672000000000000
69=171122452428141311372468338881272839092270544893520369393648040923257279754140647424000000000000000 to be followed by no of multiplications, time taken, memory used

2. Write an efficient program to print the series of first n factorial numbers starting from 1, 2, 3…. n
Input file
20
Output file
1=1
2=2
3=6
4=24
5=120
6=720
7=5040
8=40320
9=362880
10=3628800
11=39916800
12=479001600
13=6227020800
14=87178291200
15=1307674368000
16=20922789888000
17=355687428096000
18=6402373705728000
19=121645100408832000
20=2432902008176640000
Followed by no of multiplications, time taken, memory used

3. Write an efficient program to find the factorial of given n numbers using sterling formula. Fist number in the file gives value of n.
Input file
3

37
90
99
Output file
37=13763753091226345046315979581580902400000000
90=14857159644817614973095227336208257378855699612846887669422168637049853930940658765459921313708840596456172344699781120000000000000000000000
99=933262154439441526816992388562667004907159682643816214685929638952175999932299156089414639761565182862536979208272237582511852109168640000000000000000000000
Followed by no of multiplications, time taken, memory used

4. Write an efficient program to find the factorial of a number without using multiplication.
Input file
17
Output file
17=355687428096000 followed by no of additions, time taken and memory used

5. Write an efficient program to find the double factorial of k written as k!!.
Input file
9
Output file
9=945 followed by no of multiplications, time taken, memory used

6. Write an efficient program to find $\frac{n!k!}{r!}$ where value of n, k, r are given. First Number represents number of such terms.
Input file
3
7
10
4
21
4
6
7
9
10
Output file
762048000
17030314057236480000
504 followed by no of multiplications, time taken, memory used

7. A multifactorial is a generalization of the double factorial e.g. n!!!!= n(n-4)(n-8)(n-12)….till any positive number. Write an efficient program if we have the value of n and value of k where k is the number of multifactorial which is to be subtracted every time from n. First number gives number of such sets given.
Input file
2
34
4
56
7
Output file
17643225600
232436776320 followed by no of multiplications, time taken, Memory Used

8. Write an efficient program if given a set of n positive integer number, whether the given number is factorial of some other number or not. First number represents n.
Input File
4
2585201673888497664000 0
40320
43567128
91281206
Output file
1
1
0
0 followed by no of multiplications, time taken and memory used

9. Write an efficient program to find out the sum of first n factorials.
Input file
13
Output file
6749977113 followed by no of multiplications, additions, time taken, memory used

10. Write an efficient program to find out the product of first n factorials.
Input file
7
Output file
125411328000 followed by no of multiplications, time taken, memory used

Section 2

Fibonacci Numbers

1. Write an efficient program to find out and print the Fibonacci number of at the positions given in a set of n numbers given in the list. First number in the file represents n.

Input File
8
5
12
20
37
49
53
69
188

Output File
5=5
12=144
20=6765
37=24157817
49=7778742049
53=53316291173
69=117669030460994
188=87134745051736835281661581088261548838 1 followed by no of additions, time taken, memory used

2. Write an efficient program to print the series of first n Fibonacci numbers starting from 1,2,3…. n

Input file
25

Output file
1=1
2=1
3=2
4=3
5=5
6=8
7=13
8=21
9=34
10=55
11=89
12=144
13=233
14=377
15=610
16=987
17=1597
18=2584
19=4181
20=6765
21=10946
22=17711
23=28657
24=46368
25=75025 followed by no of additions, time taken, memory used

3. Write an efficient program to find the Fibonacci number of given n numbers using golden ratio formula. First number represents n.
Input file
3
37
99
167
Output file
37=24157817
99=218922995834555169026
167=356000755459584589632228765813167 53 followed by no of multiplications, time taken, Memory Used

4. Write an efficient program to find the Fibonacci value of a number using matrix multiplication.
Input file
n
15
Output file
15=610 followed by number of multiplications, time taken, memory used

5. Write an efficient program to find the first n Fibonacci numbers which are prime also.
Input file
12
Output file
3
4
5
7
11
13
17
23
29
43
47
83 followed by number of additions, multiplications and divisions, time taken, Memory Used

6. Write an efficient program to find factorial of first n Fibonacci primes.
Input file
4
Output file
2
6
120
6227020800
followed by no of additions, multiplications, time taken, memory used

7. Write an efficient program to print Fibonacci number corresponding to the first n numbers in the Fibonacci series.
Input file
10
Output file
1
1
1

2
5
21
233
10946
5702887
139583862445 followed by no of additions, time taken, memory used

8. Write an efficient program given a set of n positive integers, find that whether the number is one of the Fibonacci numbers. First number represents n
Input file
4
6765
28657
15329
368139
Output file
1
1
0
0 followed by no of additions, time taken, memory used

9. Write an efficient program to print the sum of first n Fibonacci numbers.
Input file
19
Output file
10945 followed by no of additions, time taken, Memory Used

10. Write an efficient program to print the product of first n Fibonacci numbers.
Input file
11
Output file
10904493600 followed by no of additions, multiplications, time taken, memory used

Section 3

Prime Numbers

1. Write an efficient program to print first n prime numbers.
Input file
200
Output file
2
3
5
7
11
13
17
19
23
29
31
37
41
43
47
53
59
61
67
71
73
79
83
89
97
101
103
107
109
113
127
131
137
139
149
151
157
163
167
173
179
181
191
193
197
199
211
223

227
229
233
239
241
251
257
263
269
271
277
281
283
293
307
311
313
317
331
337
347
349
353
359
367
373
379
383
389
397
401
409
419
421
431
433
439
443
449
457
461
463
467
479
487
491
499
503
509
521
523
541
547
557
563

569
571
577
587
593
599
601
607
613
617
619
631
641
643
647
653
659
661
673
677
683
691
701
709
719
727
733
739
743
751
757
761
769
773
787
797
809
811
821
823
827
829
839
853
857
859
863
877
881
883
887
907
911
919
929

937
941
947
953
967
971
977
983
991
997
1009
1013
1019
1021
1031
1033
1039
1049
1051
1061
1063
1069
1087
1091
1093
1097
1103
1109
1117
1123
1129
1151
1153
1163
1171
1181
1187
1193
1201
1213
1217
1223 followed by no of divisions, time taken, memory used

2. Write an efficient program, given a set of n positive integers, find that whether the number is one of the prime numbers or not. First number is n.
Input File
4
514229
757
871341
987213
Output File
1
1
0

0 followed by no of divisions, time taken, Memory Used

3. Write an efficient program for writing first n prime numbers which are Fibonacci numbers also.
Input file
10
Output file
2
3
5
13
89
233
1597
28657
514229
433494437 followed by no of divisions, additions, time taken, Memory Used

4. Write an efficient program for writing all primes below a given number.
Input file
21380
Output file
2,3,5,7,11,13,17,19,23,29,31,37,41,43,47,53,59,61,67,71,73,79,83,89,97,101,103,107,109,113,127,131,137,
139,149,151,157,163,167,173,179,181,191,193,197,199,211,223,227,229,233,239,241,251,257,263,269,271
,277,281,283,293,307,311,313,317,331,337,347,349,353,359,367,373,379,383,389,397,401,409,419,421,43
1,433,439,443,449,457,461,463,467,479,487,491,499,503,509,521,523,541,547,557,563,569,571,577,587,5
93,599,601,607,613,617,619,631,641,643,647,653,659,661,673,677,683,691,701,709,719,727,733,739,743,
751,757,761,769,773,787,797,809,811,821,823,827,829,839,853,857,859,863,877,881,883,887,907,911,919
,929,937,941,947,953,967,971,977,983,991,997,1009,1013,1019,1021,1031,1033,1039,1049,1051,1061,106
3,1069,1087,1091,1093,1097,1103,1109,1117,1123,1129,1151,1153,1163,1171,1181,1187,1193,1201,1213,
1217,1223,1229,1231,1237,1249,1259,1277,1279,1283,1289,1291,1297,1301,1303,1307,1319,1321,1327,1
361,1367,1373,1381,1399,1409,1423,1427,1429,1433,1439,1447,1451,1453,1459,1471,1481,1483,1487,14
89,1493,1499,1511,1523,1531,1543,1549,1553,1559,1567,1571,1579,1583,1597,1601,1607,1609,1613,161
9,1621,1627,1637,1657,1663,1667,1669,1693,1697,1699,1709,1721,1723,1733,1741,1747,1753,1759,1777,
1783,1787,1789,1801,1811,1823,1831,1847,1861,1867,1871,1873,1877,1879,1889,1901,1907,1913,1931,1
933,1949,1951,1973,1979,1987,1993,1997,1999,2003,2011,2017,2027,2029,2039,2053,2063,2069,2081,20
83,2087,2089,2099,2111,2113,2129,2131,2137,2141,2143,2153,2161,2179,2203,2207,2213,2221,2237,223
9,2243,2251,2267,2269,2273,2281,2287,2293,2297,2309,2311,2333,2339,2341,2347,2351,2357,2371,2377,
2381,2383,2389,2393,2399,2411,2417,2423,2437,2441,2447,2459,2467,2473,2477,2503,2521,2531,2539,2
543,2549,2551,2557,2579,2591,2593,2609,2617,2621,2633,2647,2657,2659,2663,2671,2677,2683,2687,26
89,2693,2699,2707,2711,2713,2719,2729,2731,2741,2749,2753,2767,2777,2789,2791,2797,2801,2803,281
9,2833,2837,2843,2851,2857,2861,2879,2887,2897,2903,2909,2917,2927,2939,2953,2957,2963,2969,2971,
2999,3001,3011,3019,3023,3037,3041,3049,3061,3067,3079,3083,3089,3109,3119,3121,3137,3163,3167,3
169,3181,3187,3191,3203,3209,3217,3221,3229,3251,3253,3257,3259,3271,3299,3301,3307,3313,3319,33
23,3329,3331,3343,3347,3359,3361,3371,3373,3389,3391,3407,3413,3433,3449,3457,3461,3463,3467,346
9,3491,3499,3511,3517,3527,3529,3533,3539,3541,3547,3557,3559,3571,3581,3583,3593,3607,3613,3617,
3623,3631,3637,3643,3659,3671,3673,3677,3691,3697,3701,3709,3719,3727,3733,3739,3761,3767,3769,3
779,3793,3797,3803,3821,3823,3833,3847,3851,3853,3863,3877,3881,3889,3907,3911,3917,3919,3923,39
29,3931,3943,3947,3967,3989,4001,4003,4007,4013,4019,4021,4027,4049,4051,4057,4073,4079,4091,409
3,4099,4111,4127,4129,4133,4139,4153,4157,4159,4177,4201,4211,4217,4219,4229,4231,4241,4243,4253,
4259,4261,4271,4273,4283,4289,4297,4327,4337,4339,4349,4357,4363,4373,4391,4397,4409,4421,4423,4
441,4447,4451,4457,4463,4481,4483,4493,4507,4513,4517,4519,4523,4547,4549,4561,4567,4583,4591,45
97,4603,4621,4637,4639,4643,4649,4651,4657,4663,4673,4679,4691,4703,4721,4723,4729,4733,4751,475
9,4783,4787,4789,4793,4799,4801,4813,4817,4831,4861,4871,4877,4889,4903,4909,4919,4931,4933,4937,
4943,4951,4957,4967,4969,4973,4987,4993,4999,5003,5009,5011,5021,5023,5039,5051,5059,5077,5081,5
087,5099,5101,5107,5113,5119,5147,5153,5167,5171,5179,5189,5197,5209,5227,5231,5233,5237,5261,52

73,5279,5281,5297,5303,5309,5323,5333,5347,5351,5381,5387,5393,5399,5407,5413,5417,5419,5431,543
7,5441,5443,5449,5471,5477,5479,5483,5501,5503,5507,5519,5521,5527,5531,5557,5563,5569,5573,5581,
5591,5623,5639,5641,5647,5651,5653,5657,5659,5669,5683,5689,5693,5701,5711,5717,5737,5741,5743,5
749,5779,5783,5791,5801,5807,5813,5821,5827,5839,5843,5849,5851,5857,5861,5867,5869,5879,5881,58
97,5903,5923,5927,5939,5953,5981,5987,6007,6011,6029,6037,6043,6047,6053,6067,6073,6079,6089,609
1,6101,6113,6121,6131,6133,6143,6151,6163,6173,6197,6199,6203,6211,6217,6221,6229,6247,6257,6263,
6269,6271,6277,6287,6299,6301,6311,6317,6323,6329,6337,6343,6353,6359,6361,6367,6373,6379,6389,6
397,6421,6427,6449,6451,6469,6473,6481,6491,6521,6529,6547,6551,6553,6563,6569,6571,6577,6581,65
99,6607,6619,6637,6653,6659,6661,6673,6679,6689,6691,6701,6703,6709,6719,6733,6737,6761,6763,677
9,6781,6791,6793,6803,6823,6827,6829,6833,6841,6857,6863,6869,6871,6883,6899,6907,6911,6917,6947,
6949,6959,6961,6967,6971,6977,6983,6991,6997,7001,7013,7019,7027,7039,7043,7057,7069,7079,7103,7
109,7121,7127,7129,7151,7159,7177,7187,7193,7207,7211,7213,7219,7229,7237,7243,7247,7253,7283,72
97,7307,7309,7321,7331,7333,7349,7351,7369,7393,7411,7417,7433,7451,7457,7459,7477,7481,7487,748
9,7499,7507,7517,7523,7529,7537,7541,7547,7549,7559,7561,7573,7577,7583,7589,7591,7603,7607,7621,
7639,7643,7649,7669,7673,7681,7687,7691,7699,7703,7717,7723,7727,7741,7753,7757,7759,7789,7793,7
817,7823,7829,7841,7853,7867,7873,7877,7879,7883,7901,7907,7919,7927,7933,7937,7949,7951,7963,79
93,8009,8011,8017,8039,8053,8059,8069,8081,8087,8089,8093,8101,8111,8117,8123,8147,8161,8167,817
1,8179,8191,8209,8219,8221,8231,8233,8237,8243,8263,8269,8273,8287,8291,8293,8297,8311,8317,8329,
8353,8363,8369,8377,8387,8389,8419,8423,8429,8431,8443,8447,8461,8467,8501,8513,8521,8527,8537,8
539,8543,8563,8573,8581,8597,8599,8609,8623,8627,8629,8641,8647,8663,8669,8677,8681,8689,8693,86
99,8707,8713,8719,8731,8737,8741,8747,8753,8761,8779,8783,8803,8807,8819,8821,8831,8837,8839,884
9,8861,8863,8867,8887,8893,8923,8929,8933,8941,8951,8963,8969,8971,8999,9001,9007,9011,9013,9029,
9041,9043,9049,9059,9067,9091,9103,9109,9127,9133,9137,9151,9157,9161,9173,9181,9187,9199,9203,9
209,9221,9227,9239,9241,9257,9277,9281,9283,9293,9311,9319,9323,9337,9341,9343,9349,9371,9377,93
91,9397,9403,9413,9419,9421,9431,9433,9437,9439,9461,9463,9467,9473,9479,9491,9497,9511,9521,953
3,9539,9547,9551,9587,9601,9613,9619,9623,9629,9631,9643,9649,9661,9677,9679,9689,9697,9719,9721,
9733,9739,9743,9749,9767,9769,9781,9787,9791,9803,9811,9817,9829,9833,9839,9851,9857,9859,9871,9
883,9887,9901,9907,9923,9929,9931,9941,9949,9967,9973,10007,10009,10037,10039,10061,10067,10069,
10079,10091,10093,10099,10103,10111,10133,10139,10141,10151,10159,10163,10169,10177,10181,1019
3,10211,10223,10243,10247,10253,10259,10267,10271,10273,10289,10301,10303,10313,10321,10331,103
33,10337,10343,10357,10369,10391,10399,10427,10429,10433,10453,10457,10459,10463,10477,10487,10
499,10501,10513,10529,10531,10559,10567,10589,10597,10601,10607,10613,10627,10631,10639,10651,1
0657,10663,10667,10687,10691,10709,10711,10723,10729,10733,10739,10753,10771,10781,10789,10799,
10831,10837,10847,10853,10859,10861,10867,10883,10889,10891,10903,10909,10937,10939,10949,1095
7,10973,10979,10987,10993,11003,11027,11047,11057,11059,11069,11071,11083,11087,11093,11113,111
17,11119,11131,11149,11159,11161,11171,11173,11177,11197,11213,11239,11243,11251,11257,11261,11
273,11279,11287,11299,11311,11317,11321,11329,11351,11353,11369,11383,11393,11399,11411,11423,1
1437,11443,11447,11467,11471,11483,11489,11491,11497,11503,11519,11527,11549,11551,11579,11587,
11593,11597,11617,11621,11633,11657,11677,11681,11689,11699,11701,11717,11719,11731,11743,1177
7,11779,11783,11789,11801,11807,11813,11821,11827,11831,11833,11839,11863,11867,11887,11897,119
03,11909,11923,11927,11933,11939,11941,11953,11959,11969,11971,11981,11987,12007,12011,12037,12
041,12043,12049,12071,12073,12097,12101,12107,12109,12113,12119,12143,12149,12157,12161,12163,1
2197,12203,12211,12227,12239,12241,12251,12253,12263,12269,12277,12281,12289,12301,12323,12329,
12343,12347,12373,12377,12379,12391,12401,12409,12413,12421,12433,12437,12451,12457,12473,1247
9,12487,12491,12497,12503,12511,12517,12527,12539,12541,12547,12553,12569,12577,12583,12589,126
01,12611,12613,12619,12637,12641,12647,12653,12659,12671,12689,12697,12703,12713,12721,12739,12
743,12757,12763,12781,12791,12799,12809,12821,12823,12829,12841,12853,12889,12893,12899,12907,1
2911,12917,12919,12923,12941,12953,12959,12967,12973,12979,12983,13001,13003,13007,13009,13033,
13037,13043,13049,13063,13093,13099,13103,13109,13121,13127,13147,13151,13159,13163,13171,1317
7,13183,13187,13217,13219,13229,13241,13249,13259,13267,13291,13297,13309,13313,13327,13331,133
37,13339,13367,13381,13397,13399,13411,13417,13421,13441,13451,13457,13463,13469,13477,13487,13
499,13513,13523,13537,13553,13567,13577,13591,13597,13613,13619,13627,13633,13649,13669,13679,1
3681,13687,13691,13693,13697,13709,13711,13721,13723,13729,13751,13757,13759,13763,13781,13789,
13799,13807,13829,13831,13841,13859,13873,13877,13879,13883,13901,13903,13907,13913,13921,1393
1,13933,13963,13967,13997,13999,14009,14011,14029,14033,14051,14057,14071,14081,14083,14087,141

07,14143,14149,14153,14159,14173,14177,14197,14207,14221,14243,14249,14251,14281,14293,14303,14321,14323,14327,14341,14347,14369,14387,14389,14401,14407,14411,14419,14423,14431,14437,14447,14449,14461,14479,14489,14503,14519,14533,14537,14543,14549,14551,14557,14561,14563,14591,14593,14621,14627,14629,14633,14639,14653,14657,14669,14683,14699,14713,14717,14723,14731,14737,14741,14747,14753,14759,14767,14771,14779,14783,14797,14813,14821,14827,14831,14843,14851,14867,14869,14879,14887,14891,14897,14923,14929,14939,14947,14951,14957,14969,14983,15013,15017,15031,15053,15061,15073,15077,15083,15091,15101,15107,15121,15131,15137,15139,15149,15161,15173,15187,15193,15199,15217,15227,15233,15241,15259,15263,15269,15271,15277,15287,15289,15299,15307,15313,15319,15329,15331,15349,15359,15361,15373,15377,15383,15391,15401,15413,15427,15439,15443,15451,15461,15467,15473,15493,15497,15511,15527,15541,15551,15559,15569,15581,15583,15601,15607,15619,15629,15641,15643,15647,15649,15661,15667,15671,15679,15683,15727,15731,15733,15737,15739,15749,15761,15767,15773,15787,15791,15797,15803,15809,15817,15823,15859,15877,15881,15887,15889,15901,15907,15913,15919,15923,15937,15959,15971,15973,15991,16001,16007,16033,16057,16061,16063,16067,16069,16073,16087,16091,16097,16103,16111,16127,16139,16141,16183,16187,16189,16193,16217,16223,16229,16231,16249,16253,16267,16273,16301,16319,16333,16339,16349,16361,16363,16369,16381,16411,16417,16421,16427,16433,16447,16451,16453,16477,16481,16487,16493,16519,16529,16547,16553,16561,16567,16573,16603,16607,16619,16631,16633,16649,16651,16657,16661,16673,16691,16693,16699,16703,16729,16741,16747,16759,16763,16787,16811,16823,16829,16831,16843,16871,16879,16883,16889,16901,16903,16921,16927,16931,16937,16943,16963,16979,16981,16987,16993,17011,17021,17027,17029,17033,17041,17047,17053,17077,17093,17099,17107,17117,17123,17137,17159,17167,17183,17189,17191,17203,17207,17209,17231,17239,17257,17291,17293,17299,17317,17321,17327,17333,17341,17351,17359,17377,17383,17387,17389,17393,17401,17417,17419,17431,17443,17449,17467,17471,17477,17483,17489,17491,17497,17509,17519,17539,17551,17569,17573,17579,17581,17597,17599,17609,17623,17627,17657,17659,17669,17681,17683,17707,17713,17729,17737,17747,17749,17761,17783,17789,17791,17807,17827,17837,17839,17851,17863,17881,17891,17903,17909,17911,17921,17923,17929,17939,17957,17959,17971,17977,17981,17987,17989,18013,18041,18043,18047,18049,18059,18061,18077,18089,18097,18119,18121,18127,18131,18133,18143,18149,18169,18181,18191,18199,18211,18217,18223,18229,18233,18251,18253,18257,18269,18287,18289,18301,18307,18311,18313,18329,18341,18353,18367,18371,18379,18397,18401,18413,18427,18433,18439,18443,18451,18457,18461,18481,18493,18503,18517,18521,18523,18539,18541,18553,18583,18587,18593,18617,18637,18661,18671,18679,18691,18701,18713,18719,18731,18743,18749,18757,18773,18787,18793,18797,18803,18839,18859,18869,18899,18911,18913,18917,18919,18947,18959,18973,18979,19001,19009,19013,19031,19037,19051,19069,19073,19079,19081,19087,19121,19139,19141,19157,19163,19181,19183,19207,19211,19213,19219,19231,19237,19249,19259,19267,19273,19289,19301,19309,19319,19333,19373,19379,19381,19387,19391,19403,19417,19421,19423,19427,19429,19433,19441,19447,19457,19463,19469,19471,19477,19483,19489,19501,19507,19531,19541,19543,19553,19559,19571,19577,19583,19597,19603,19609,19661,19681,19687,19697,19699,19709,19717,19727,19739,19751,19753,19759,19763,19777,19793,19801,19813,19819,19841,19843,19853,19861,19867,19889,19891,19913,19919,19927,19937,19949,19961,19963,19973,19979,19991,19993,19997,20011,20021,20023,20029,20047,20051,20063,20071,20089,20101,20107,20113,20117,20123,20129,20143,20147,20149,20161,20173,20177,20183,20201,20219,20231,20233,20249,20261,20269,20287,20297,20323,20327,20333,20341,20347,20353,20357,20359,20369,20389,20393,20399,20407,20411,20431,20441,20443,20477,20479,20483,20507,20509,20521,20533,20543,20549,20551,20563,20593,20599,20611,20627,20639,20641,20663,20681,20693,20707,20717,20719,20731,20743,20747,20749,20753,20759,20771,20773,20789,20807,20809,20849,20857,20873,20879,20887,20897,20899,20903,20921,20929,20939,20947,20959,20963,20981,20983,21001,21011,21013,21017,21019,21023,21031,21059,21061,21067,21089,21101,21107,21121,21139,21143,21149,21157,21163,21169,21179,21187,21191,21193,21211,21221,21227,21247,21269,21277,21283,21313,21317,21319,21323,21341,21347,21377,21379
followed by no of divisions, time taken, Memory Used

5. Write an efficient program for writing first prime number greater than a given number.
Input file
100000
Output file
100003 followed by no of divisions, time taken, Memory Used

6. Write an efficient program to find out the prime factors of a given set of n numbers. First number gives number of n.
Input file
5
48
3750
256
5901
7654321
Output file
2
2
2
2
3

2
3
5
5
5
5

2
2
2
2
2
2
2
2

3
7
281

19
402859 followed by number of divisions, time taken, Memory Used

7. Primorial numbers are factorials of prime numbers. Write an efficient program to find out the factorial of first n prime numbers which will generate a series of first n primorials.
Input file
10
Output file
2
6
120
5040
39916800
6227020800
355687428096000
121645100408832000
2585201673888497664000
8841761993739701954543616000000 followed by no of divisions, multiplications, time taken, Memory Used

8. Write an efficient program to find the sum of first n prime numbers.
Input file
100
Output file
24133 followed by no of divisions, time taken, Memory Used

9. Write an efficient program to find the product of first n prime numbers.
Input file
40
Output file
166589903787325219380851695350896256250980509594874862046961683989710 followed by no of divisions, multiplications, time taken, Memory Used

10. Write an efficient program to print the nth prime of the first m numbers of prime series.
For example first prime is 2 so print 2^{nd} prime, second prime is 3 so print 3^{rd} prime and so on.
Input file
10
Output file
3
5
11
17
31
41
59
67
83
109 followed by no of divisions, time taken, Memory Used

Section 4

Powers of a Number

1. Given a number m, write all the values corresponding to the first n Fibonacci exponents of that number.
Input file
6
9
Output file
6
6
36
216
7776
1679616
13060694016
21936950640377856
28651179995807043183810969 6 followed by no of additions, multiplications time taken, memory used

2. Given a number m, write all the values corresponding to the first n prime exponents of that number.
Input file
4
10
Output file
16
64
1024
16384
4194304
67108864
17179869184
274877906944
70368744177664
288230376151711744 followed by no of divisions, multiplications time taken, memory used

3. Given a number and an exponent write all the values of powers of that number from one to given number.
Input file
7
11
Output file
7
49
343
2401
16807
117649
823543
5764801
40353607
282475249
1977326743 followed by no of multiplications time taken, Memory Used

4. Given two numbers x and y, find the value of x exponent y (or x power y).
Input file
5
25
Output file

298023223876953125 followed by no of multiplications time taken, Memory Used

5. Given two numbers x and y find $x^y + y^x$ and $x^y - y^x$ and x^y / y^x and $x^y * y^x$
Input file
6
8
Output file
1941760
1417472
6.407
440301256704 followed by no of additions, multiplications time taken, memory used

6. Given a number n, find the sum of numbers from 1 to n.
Input file
500
Output file
125250 followed by no of additions, multiplications time taken, Memory Used

7. Given a number n, find the sum of squares of numbers from 1 to n.
Input file
75
Output file
143450 followed by no of additions, multiplications time taken, memory used

8. Given a number n, find the sum of cubes of numbers from 1 to n.
Input file
39
Output file
608400 followed by no of additions, multiplications time taken, memory used

9. Given a number n, find the sum of inverse of numbers from 1 to n.
Input file
1000
Output file
7.485 followed by no of additions, multiplications time taken, memory used

10. Given a number n, find the sum of series of exponents of 2 from 1 to n.
Input file
40
Output file
2199023255550
followed by no of additions, multiplications time taken, memory used

Section 5

Number Conversion

1. Write an efficient program to find the octal equivalent of a series of decimal numbers from m to n.

Input file

23
99

Output file

27
30
31
32
33
34
35
36
37
40
41
42
43
44
45
46
47
51
52
53
54
55
56
57
60
61
62
63
64
65
66
67
70
71
72
73
74
75
76
77
100
101
102
103
104
105
106
107

110
111
112
113
114
115
116
117
120
121
122
123
124
125
126
127
130
131
132
133
134
135
136
137
138
139
140
141
142
143 followed by no of additions, multiplications time taken, Memory Used

2. Write an efficient program to find the octal equivalent of a series of binary numbers from m to n.
Input file
1111
11000
Output file
17
20
21
22
23
24
25
26
27
30 followed by no of additions, multiplications time taken, Memory Used

3. Write an efficient program to find the octal equivalent of a series of hexadecimal numbers from m to n.
Input file
1A
2A
Output File
32
33
34

35
36
37
40
41
42
43
44
45
46
47
51
52 followed by no of additions, multiplications time taken, memory used

4. Write an efficient program to find the hexadecimal equivalent of a series of binary number from m to n.
Input file
10000000
10001000
Output File
80
81
82
83
84
85
86
87
88 followed by no of additions, multiplications time taken and memory used

5. Write an efficient program to find the hexadecimal equivalent of a given series of decimal number from m to n.
Input file
401
413
Output file
191
192
193
194
195
196
197
198
199
19A
19B
19C
19D followed by no of additions, multiplications time taken, memory used

6. Write an efficient program to find the hexadecimal equivalent of a given series of octal numbers from m to n.
Input file
127
137
Output file

57
58
59
5A
5B
5C
5D
5E
5F followed by no of additions, multiplications time taken, memory used

7. Write an efficient program to find the decimal equivalent of a given series of binary numbers from m to n.
Input file
10000100
10001111
Output file
132
133
134
135
136
137
138
139
140
141
142
143 followed by no of additions, multiplications time taken, memory used

8. Write an efficient program to find the decimal equivalent of a given series of octal number from m to n.
Input file
1027
1041
Output file
535
536
537
538
539
540
541
542
543
544
545 followed by no of additions, multiplications time taken, memory used

9. Write an efficient program to find the decimal equivalent of a given series of hexadecimal numbers from m to n.
Input file
113A
114D
Output file
4410
4411
4412
4413

4414
4415
4416
4417
4418
4419
4420
4421
4422
4423
4424
4425
4426
4427
4428
4429 followed by no of additions, multiplications time taken, memory used

10. Write an efficient program to find the binary equivalent of a given decimal number from m to n.
Input file
10001
10019
Output file
10011100010001
10011100010010
10011100010011
10011100010100
10011100010101
10011100010110
10011100010111
10011100011000
10011100011001
10011100011010
10011100011011
10011100011100
10011100011101
10011100011110
10011100011111
10011100100000
10011100100001
10011100100010
10011100100011 followed by no of additions, multiplications time taken, Memory Used

Section 6

Basic Math

1. Write an efficient program to find out the root of quadratic equation with coefficients as a,b and c.
Input file
5
6
1
Output file
-0.2
-1 followed by no of additions, multiplications time taken, memory used

2. Write an efficient program to find out the Greatest common divisor using recursion for all values of (m, n) where m varies from x to y and n varies from p to q
Input file
12345
12355
222
232
Output file
3
1
1
9
1
1
3
1
1
3
1 followed by no of additions, multiplications time taken, memory used

3. Write an efficient program to find the binary equivalent of a given series of octal numbers from m to n.
Input file
11127
11137
Output file
1001001010111
1001001011000
1001001011001
1001001011010
1001001011011
1001001011100
1001001011101
1001001011110
1001001011111 followed by no of additions, multiplications time taken, Memory Used
n
4. Write an efficient program to find the binary equivalent of a given series of hexadecimal numbers from m to n.
Input file
ABCEA
ABCF5
Output file
10101011110011101010
10101011110011101011
10101011110011101100

10101011110011101101
10101011110011101110
10101011110011101111
10101011110011110000
10101011110011110001
10101011110011110010
10101011110011110011
10101011110011110100
10101011110011110101 Followed by no of additions, multiplications time taken, Memory Used

5. Write an efficient program to find out the compound interest if rate of interest, principal amount, number of years is given.
Input file
8
3452100
4
Output file
1244443.94 followed by no of additions, multiplications time taken, memory used

6. Given a number print the table of nth number from 1 to 10 in a proper format.
Input file
20
Output file
1*20=20
2*20=40
3*20=60
4*20=80
5*20=100
6*20=120
7*20=140
8*20=160
9*20=180
10*20=200 followed by no of additions, multiplications time taken, Memory Used

7. Given a number print n sequences of m each such that the sequences are of the form
n,n+1,n+2….n+m
n,n+2,n+4….n+2m
n,n+3,n+6….n+3m
……..
Input file
10
10
Output file
10,11,12,13,14,15,16,17,18,19
10,12,14,16,18,20,22,24,26,28
10,13,16,19,22,25,28,31,34,37
10,14,18,22,26,30,34,38,42,46
10,15,20,25,30,35,40,45,50,55
10,16,22,28,34,40,46,52,58,64
10,17,24,31,38,45,52,59,66,73
10,18,26,34,42,50,58,66,74,82
10,19,28,37,46,55,64,73,82,91
10,20,30,40,50,60,70,80,90,100 followed by no of additions, multiplications time taken, Memory Used

8. Write an efficient program if given a number x, find out the sum of the first n terms of geometric series $1+x+x^2+x^3+x^4+\ldots..$
Input file
3
11
Output file
88573 followed by no of additions, multiplications time taken, memory used

9. Write an efficient program if given a number x, find out the sum of n terms of arithmetic progression $x+(x+d)+(x+2d)+(x+3d)\ldots\ldots..$
Input file as(x,d,n)
10
9
100
Output file
45550 followed by no of additions, multiplications time taken, Memory Used

10. Write an efficient program to find the floor of the mean of given set of n numbers. First number gives n.
Input file
13
23
43
56
76
85
92
16
61
41
89
27
53
28
Output file
53 followed by no of additions, multiplications time taken, Memory Used

Section 7

Dealing with numbers

1. Write an efficient program to find the maximum of a given set of n numbers. First number gives n.
Input file
20,428, 313, 629, 956, 556, 815, 46, 257, 603, 501, 923, 755, 192, 150, 511, 563, 986, 312, 515, 858
Output file
986 followed by no of comparisons time taken, Memory Used

2. Write an efficient program to print the sum of the individual digits of numbers from m to n.
Input file
12453
12462
Output file
15
16
17
18
19
20
21
13
14
15 followed by no of additions, multiplications time taken, Memory Used

3. Write an efficient program to print the reverse of a given numbers from m to n.
Input file
978, 1000
Output file
879
979
089
189
289
389
489
589
689
789
889
989
099
199
299
399
499
599
699
799
899
999
0001
followed by no of additions, multiplications time taken, Memory Used

4. Write an efficient program to find which numbers from range m to n are perfect numbers. A perfect number is a positive integer that is equal to the sum of it divisors. However, for the case of a perfect number, the number itself is not included in the sum. For example divisors of 6 are 1,2 and 3 and their sum is 6.
Input file
1
500
Output file
6
28
496 followed by no of additions, multiplications time taken, Memory Used

5. Write an efficient program to find whether a given set of n numbers is a triangular number or not. first number is n.
Input file
5
21
55
112
561
8765
Output file
1
1
0
1
0 followed by no of additions, multiplications time taken, Memory Used

6. Write an efficient program to find the smallest and the next smallest number in an array. First number gives the size.
Input file
20
428
313
629
956
556
815
46
257
603
501
923
755
192
150
511
563
986
312
515
858
Output file
46
150 followed by no of additions, multiplications time taken, Memory Used

7. Write an efficient program to find whether a given number is present in the given list of numbers or not. First number is the number to be found from the remaining list of numbers.
Input file
705,428
313
629
956
556
815
46
257
603
501
923
755
192
150
511
563
986
312
515
858
Output file
0 followed by no of comparisons time taken, Memory Used

8. Write an efficient program to convert a number to sum of the ASCII values of its digits.
Input file
9871402336182
Output file
678 followed by no of additions, multiplications time taken, Memory Used

9. Write an efficient program to do bitwise ANDing of two given binary numbers.
Input file
1000111010101110011
1011000011111010101011
Output file
000000000111000000100011 followed by no of additions, multiplications time taken, Memory Used

10. Write an efficient program to do bitwise ORing of two given numbers.
Input file
1000111010101110011
1011000011111010101011
Output file
101101001111111111111011 Followed by no of additions, multiplications time taken, Memory Used

Section-8

Arrays

1. An array of size n consists of unique records where the record consists of numbers consisting of three digits from 500 to 999. Insert m numbers from 1 to 100 at given positions. Count the total number of shifts in your program. Print time taken, Memory Used. Also print the size of the array and the array itself at the end of the program. First Number gives n and second gives m

Input File

100
100

Initial values in the array

971
951
551
590
824
932
669
574
655
985
873
914
855
906
689
751
502
608
815
790
775
532
754
688
650
846
606
763
608
795
890
894
819
577
705
710
560
641
773
811
585
563
700
996
512

541
912
624
796
508
842
729
655
579
692
827
873
697
623
744
561
839
961
941
599
799
582
532
505
660
773
879
925
738
896
685
660
645
960
683
927
515
912
959
944
977
874
867
705
907
625
715
502
919
842
851
891
847
585
689

Second Input File containing positions for insertions
33
12
99
12
54
45
96
93
90
100
43
24
13
77
17
46
40
99
35
3
97
22
97
43
56
23
60
99
14
3
86
73
10
51
47
60
45
43
34
64
62
15
95
58
90
75
8
77
72
47
8
45
1
16

47
83
57
46
25
56
9
60
42
56
38
39
45
10
74
53
97
82
43
13
68
24
94
49
87
45
83
95
61
69
15
51
72
37
16
81
79
47
87
39
30
8
65
78
48
71

Output File

2. An array of size n consists of unique records where the record consists of numbers consisting of three digits from 500 to 999. Delete m numbers at given positions. Count the total number of shifts in your program. Print time taken, Memory Used. Also print the size of the array and the array itself at the end of the program. First Number gives n and second gives m

Input File

200
100

First input file containing initial array
571
514
565
805
862
649
817
543
641
993
503
759
879
965
973
952
733
564
651
940
926
671
655
663
997
672
927
836
885
662
798
960
892
856
995
945
768
579
523
598
824
559
985
564
976
594
719
962
935
995
634
890
516
949

742
588
598
778
620
705
976
680
789
718
621
601
530
746
972
888
798
771
893
853
881
535
676
816
919
654
977
882
664
704
604
746
784
525
702
953
844
607
903
883
791
930
547
826
656
729
709
634
999
730
748
904
905
506
804

697
736
763
940
848
931
724
560
648
528
953
780
664
899
901
833
788
752
629
908
785
567
505
698
716
574
614
538
773
897
803
874
517
778
882
864
887
958
578
532
570
713
859
888
630
859
650
710
811
752
875
522
912
794
734

903
596
955
833
754
558
563
521
817
598
947
567
877
817
770
864
942
655
799
859
771
759
519
760
796
947
530
844
794
846
791
541
844
903
901
534
Second input file containing positions for deletions
33
12
99
12
54
45
96
93
90
100
43
24
13
77
17
46
40
99

35
3
97
22
97
43
56
23
60
99
14
3
86
73
10
51
47
60
45
43
34
64
62
15
95
58
90
75
8
77
72
47
8
45
1
16
47
83
57
46
25
56
9
60
42
56
38
39
45
10
74
53
97
82
43

13
68
24
94
49
87
45
83
95
61
69
15
51
72
37
16
81
79
47
87
39
30
8
65
78
48
71
Output File

3. An array of size n consists of records where the record consists of numbers consisting of three digits from 900 to 999. Delete m given numbers. All the occurrences of that number are to be deleted. Count the total number of shifts in your program. Print time taken, Memory Used. Also print the size of the array and the array itself at the end of the program. First Number gives n and second gives m
Input File
200,50
Initial array
927
913
909
998
926
953
909
929
920
976
989
984
981
965
966
978
970
925
930

902
933
952
915
904
932
983
914
916
907
900
969
941
958
915
969
970
937
997
948
971
960
931
911
960
992
948
979
900
950
900
928
937
957
999
937
905
991
920
915
998
932
982
979
927
981
922
953
901
971
934
926
992
991
915

947
958
982
982
997
979
967
990
918
962
902
968
996
966
943
948
928
919
912
974
965
941
913
958
920
992
939
957
981
903
902
923
998
916
987
976
918
919
981
959
934
923
907
930
998
939
987
958
965
996
929
927
960
947
960

981
992
967
996
926
972
936
931
956
939
948
993
943
982
911
999
984
983
948
909
978
947
964
932
993
979
908
968
908
954
964
980
971
936
991
949
927
923
927
921
908
973
928
986
935
918
920
995
946
927
971
983
911
989
974

941
940
915
954
929
940
938
977
992
954
948
917
931
992
947
982

Second input file containing numbers for deletion

996
902
943
916
937
992
979
940
987
909
965
978
999
921
982
969
918
994
913
904
958
948
900
959
992
913
961
957
921
983
995
938
933
967
941
954
936
920

915
966
966
966
914
904
927
940
990
961
920
902
Output File

4. An array of size n consists of records where the record consists of numbers consisting of three digits from 900 to 999. Insert m numbers again from the same range at given positions. Count the total number of shifts in your program. Print time taken, Memory Used. Also print the size of the array and the array itself at the end of the program. First Number gives n and second gives m
Input File
100,50
Initial array
922
925
960
974
943
916
936
953
981
946
920
910
905
980
924
960
907
973
974
984
960
980
953
991
988
968
964
923
934
927
989
945
946

904
918
975
929
973
965
970
964
956
988
950
914
993
949
925
992
934
991
979
997
939
990
956
959
918
926
939
919
989
978
936
996
904
941
972
999
952
942
937
994
948
926
987
995
922
953
997
946
927
987
952
951
969
913
919

901
907
914
957
912
995
909
903
984
939
964
935
Numbers to be inserted
907
961
937
985
904
999
909
950
919
924
988
998
911
953
974
948
956
942
938
908
926
937
960
968
917
903
908
914
979
904
950
983
973
985
939
905
963
975
905
904
916
997

919
916
958
972
995
902
952
937
Positions
17
5
9
21
1
7
30
34
32
46
20
7
16
21
8
36
20
14
4
34
26
36
35
25
18
45
11
24
2
6
23
23
10
50
41
22
50
18
26
23
30
44
21
12
39
13

12
12
23
27
Output File

5. A stack of size n (To be implemented using an array) consists of unique records where the record consists of numbers consisting of three digits from 500 to 999. Push m given numbers at given position. Also Delete m elements from the specified given position from the stack. Count the total number of push and pop in your program. Print time taken, Memory Used. Also print the size of the stack and the stack itself at the end of the program. First Number gives n and second gives m
Input File
100
50
Initial stack elements
971
951
551
590
824
932
669
574
655
985
873
914
855
906
689
751
502
608
815
790
775
532
754
688
650
846
606
763
608
795
890
894
819
577
705
710
560
641
773
811

585
563
700
996
512
541
912
624
796
508
842
729
655
579
692
827
873
697
623
744
561
839
961
941
599
799
582
532
505
660
773
879
925
738
896
685
660
645
960
683
927
515
912
959
944
977
874
867
705
907
625
715
502
919
842

851
891
847
585
689
Numbers to be be Inserted
67
47
39
47
9
32
49
23
66
41
86
51
22
82
46
79
46
45
7
94
99
46
28
33
97
30
40
28
71
2
44
84
69
92
23
16
26
19
36
27
32
99
17
30
29
89
23
57

43
51
Positions for Insertions
67
47
39
47
9
32
49
23
66
41
86
51
22
82
46
79
46
45
7
94
99
46
28
33
97
30
40
28
71
2
44
84
69
92
23
16
26
19
36
27
32
99
17
30
29
89
23
57
43
51
positions for Deletions

67
47
39
47
9
32
49
23
66
41
86
51
22
82
46
79
46
45
7
94
99
46
28
33
97
30
40
28
71
2
44
84
69
92
23
16
26
19
36
27
32
99
17
30
29
89
23
57
43
51
Output File

6. A Queue of size n (Implemented with an array) consists of unique records where the record consists of numbers consisting of three digits from 500 to 999. Insert m numbers from 1 to 100 in the queue. Each

insertion should be followed by one dequeue. Count the total number of operations in your program. Print time taken, Memory Used. Print the size of the queue and the queue itself at the end of the program. First Number gives n and second gives m

Input File

100,100

Initial Queue contents

971
951
551
590
824
932
669
574
655
985
873
914
855
906
689
751
502
608
815
790
775
532
754
688
650
846
606
763
608
795
890
894
819
577
705
710
560
641
773
811
585
563
700
996
512
541
912
624
796

508
842
729
655
579
692
827
873
697
623
744
561
839
961
941
599
799
582
532
505
660
773
879
925
738
896
685
660
645
960
683
927
515
912
959
944
977
874
867
705
907
625
715
502
919
842
851
891
847
585
689

Numbers to be inserted

33
12
99

12
54
45
96
93
90
100
43
24
13
77
17
46
40
99
35
3
97
22
97
43
56
23
60
99
14
3
86
73
10
51
47
60
45
43
34
64
62
15
95
58
90
75
8
77
72
47
8
45
1
16
47
83
57
46

25
56
9
60
42
56
38
39
45
10
74
53
97
82
43
13
68
24
94
49
87
45
83
95
61
69
15
51
72
37
16
81
79
47
87
39
30
8
65
78
48
71
Output File

7. You have an array A of size N. Write a routine that shuffles all the elements of the array in-place. The only restrictions are that all possible permutations of A must be possible and equally likely. First number given values of N.
Input file
100
33
12
99
12
54
45

96
93
90
100
43
24
13
77
17
46
40
99
35
3
97
22
97
43
56
23
60
99
14
3
86
73
10
51
47
60
45
43
34
64
62
15
95
58
90
75
8
77
72
47
8
45
1
16
47
83
57
46
25
56
9

60
42
56
38
39
45
10
74
53
97
82
43
13
68
24
94
49
87
45
83
95
61
69
15
51
72
37
16
81
79
47
87
39
30
8
65
78
48
71
Output File

8. Suppose that each row of an mXn array A consists of 1's and 0's such that in any row i of A, all the 1's come before any 0's in that row. Suppose further that the number of 1's in the row i is less than or equal to the number in row i+1, for i = 0,1,……n-2. Assuming A is already in memory, describe a method running in O(n) time for counting the number of 1's in the array.First number gives m and second gives n.
Input file
9
22
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0
1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0
1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0

1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
Output File
87

9. Given an array A of size 26X26. Determine whether each row and each column of the array consists of the set {A,B,C,…..,Z} where each element occurs exactly once.
Input File
ABCDEFGHIJKLMNOPQRSTUVWXYZ
ZABCDEFGHIJKLMNOPQRSTUVWXY
YZABCDEFGHIJKLMNOPQRSTUVWX
LMNOPQRSTUVWXYZABCDEFGHIJK
CDEFGHIJKLMNOPQRSTUVWXYZAB
MNOPQRSTUVWXYZABCDEFGHIJKL
XYZABCDEFGHIJKLMNOPQRSTUVW
DEFGHIJKLMNOPQRSTUVWXYZABC
KLMNOPQRSTUVWXYZABCDEFGHIJ
RSTUVWXYZABCDEFGHIJKLMNOPQ
WXYZABCDEFGHIJKLMNOPQRSTUV
FGHIJKLMNOPQRSTUVWXYZABCDE
EFGHIJKLMNOPQRSTUVWXYZABCD
QRSTUVWXYZABCDEFGHIJKLMNOP
VWXYZABCDEFGHIJKLMNOPQRSTU
JKLMNOPQRSTUVWXYZABCDEFGHI
GHIJKLMNOPQRSTUVQXYZABCDEF
PQRSTUVWXYZABCDEFGHIJKLMNO
UVWXYZABCDEFGHIJKLMNOPQRST
IJKLMNOPQRSTUVWXYZABCDEFGH
OPQRSTUVWXYZABCDEFGHIJKLMN
TUVWXYZABCDEFGHIJKLMNOPQRS
BCDEFGHIJKLMNOPQRSTUVWXYZA
NOPQRSTUVWXYZABCDEFGHIJKLM
HIJKLMNOPQRSTUVWXYZABCDEFG
STUVWXYZABCDEFGHIJKLMNOPQR
Output File
1

10. A certain one Way Street has m parking spaces in a row, numbered 1 through m. A man and his dozing wife drive by, and suddenly she wakes up and orders him to park immediately. He dutifully parks at the first available space, but if there are no available places he follows linear probing but does not backs up. Suppose this happens for n different cars, where the jth wife wakes up just in time to park at space a_j.check whether the given sequence can be safely parked assuming that street is initially empty and no one leaves after parking. First number is n and second number is m.
Input file
9
9
3
1
4
1
5
9
2
6
5

Output file
2
4
1
3
5
7
8
9
6

Section-9

Link Lists

1. A Link List of size m consists of unique records where the record consists of numbers consisting of three digits from 500 to 999. Insert n numbers from 1 to 100 at given position. Count the total number of operations in your program. Print time taken, Memory Used. Also print the size of the list and the list itself at the end of the program. First number gives m and second number gives n.

Input File

100,100

Initial values in the list 971

951
551
590
824
932
669
574
655
985
873
914
855
906
689
751
502
608
815
790
775
532
754
688
650
846
606
763
608
795
890
894
819
577
705
710
560
641
773
811
585
563
700
996
512
541
912

624
796
508
842
729
655
579
692
827
873
697
623
744
561
839
961
941
599
799
582
532
505
660
773
879
925
738
896
685
660
645
960
683
927
515
912
959
944
977
874
867
705
907
625
715
502
919
842
851
891
847
585
689
positions for insertions

33
12
99
12
54
45
96
93
90
100
43
24
13
77
17
46
40
99
35
3
97
22
97
43
56
23
60
99
14
3
86
73
10
51
47
60
45
43
34
64
62
15
95
58
90
75
8
77
72
47
8
45
1
16
47

83
57
46
25
56
9
60
42
56
38
39
45
10
74
53
97
82
43
13
68
24
94
49
87
45
83
95
61
69
15
51
72
37
16
81
79
47
87
39
30
8
65
78
48
71

2. A Link List of size m consists of unique records where the record consists of numbers consisting of three digits from 500 to 999. Delete n numbers at given position. Count the total number of operations in your program. Print time taken, Memory Used. Also print the size of the list and the list itself at the end of the program. First number gives m and second number gives n.

Input File

200

100

Initial Link list

571
514
565
805
862
649
817
543
641
993
503
759
879
965
973
952
733
564
651
940
926
671
655
663
997
672
927
836
885
662
798
960
892
856
995
945
768
579
523
598
824
559
985
564
976
594
719
962
935
995
634
890
516
949
742

588
598
778
620
705
976
680
789
718
621
601
530
746
972
888
798
771
893
853
881
535
676
816
919
654
977
882
664
704
604
746
784
525
702
953
844
607
903
883
791
930
547
826
656
729
709
634
999
730
748
904
905
506
804
697

736
763
940
848
931
724
560
648
528
953
780
664
899
901
833
788
752
629
908
785
567
505
698
716
574
614
538
773
897
803
874
517
778
882
864
887
958
578
532
570
713
859
888
630
859
650
710
811
752
875
522
912
794
734
903

596
955
833
754
558
563
521
817
598
947
567
877
817
770
864
942
655
799
859
771
759
519
760
796
947
530
844
794
846
791
541
844
903
901
534
positions for deletions
33
12
99
12
54
45
96
93
90
100
43
24
13
77
17
46
40
99
35

3
97
22
97
43
56
23
60
99
14
3
86
73
10
51
47
60
45
43
34
64
62
15
95
58
90
75
8
77
72
47
8
45
1
16
47
83
57
46
25
56
9
60
42
56
38
39
45
10
74
53
97
82
43
13

68
24
94
49
87
45
83
95
61
69
15
51
72
37
16
81
79
47
87
39
30
8
65
78
48
71

3. A Link list of size m consists of records where the record consists of numbers consisting of three digits from 900 to 999. Delete n given numbers. All the occurrences of that number are to be deleted. Count the total number of operations in your program. Print time taken, Memory Used. Also print the size of the list and the list itself at the end of the program. First number gives m and second number gives n.

Input File

200
50

Initial list

927
913
909
998
926
953
909
929
920
976
989
984
981
965
966
978
970
925
930
902

933
952
915
904
932
983
914
916
907
900
969
941
958
915
969
970
937
997
948
971
960
931
911
960
992
948
979
900
950
900
928
937
957
999
937
905
991
920
915
998
932
982
979
927
981
922
953
901
971
934
926
992
991
915
947

958
982
982
997
979
967
990
918
962
902
968
996
966
943
948
928
919
912
974
965
941
913
958
920
992
939
957
981
903
902
923
998
916
987
976
918
919
981
959
934
923
907
930
998
939
987
958
965
996
929
927
960
947
960
981

992
967
996
926
972
936
931
956
939
948
993
943
982
911
999
984
983
948
909
978
947
964
932
993
979
908
968
908
954
964
980
971
936
991
949
927
923
927
921
908
973
928
986
935
918
920
995
946
927
971
983
911
989
974
941

940
915
954
929
940
938
977
992
954
948
917
931
992
947
982

numbers for deletion 996
902
943
916
937
992
979
940
987
909
965
978
999
921
982
969
918
994
913
904
958
948
900
959
992
913
961
957
921
983
995
938
933
967
941
954
936
920
915

966
966
966
914
904
927
940
990
961
920
902

4. A Link list of size m consists of records where the record consists of numbers consisting of three digits from 900 to 999. Insert n numbers again from the same range at given positions. Count the total number of operations in your program. Print time taken, Memory Used. Also print the size of the list and the list itself at the end of the program. First number gives m and second number gives n.

Input File

100
50

Initial list 922
925
960
974
943
916
936
953
981
946
920
910
905
980
924
960
907
973
974
984
960
980
953
991
988
968
964
923
934
927
989
945
946
904
918
975

929
973
965
970
964
956
988
950
914
993
949
925
992
934
991
979
997
939
990
956
959
918
926
939
919
989
978
936
996
904
941
972
999
952
942
937
994
948
926
987
995
922
953
997
946
927
987
952
951
969
913
919
901
907
914

957
912
995
909
903
984
939
964
935
Numbers to be inserted
907
961
937
985
904
999
909
950
919
924
988
998
911
953
974
948
956
942
938
908
926
937
960
968
917
903
908
914
979
904
950
983
973
985
939
905
963
975
905
904
916
997
919
916
958

972
995
902
952
937
Positions for Insertion
17
5
9
21
1
7
30
34
32
46
20
7
16
21
8
36
20
14
4
34
26
36
35
25
18
45
11
24
2
6
23
23
10
50
41
22
50
18
26
23
30
44
21
12
39
13
12
12

23
27

5. A Doubly circular Link List of size m consists of unique records where the record consists of numbers consisting of three digits from 500 to 999. Insert n numbers at given positions. Count the total number of operations in your program. Print time taken, Memory Used. Also print the size of the list and the list itself at the end of the program. First number gives m and second number gives n.

Input File

100
100

Initial values in the list

971
951
551
590
824
932
669
574
655
985
873
914
855
906
689
751
502
608
815
790
775
532
754
688
650
846
606
763
608
795
890
894
819
577
705
710
560
641
773
811
585
563
700
996

512
541
912
624
796
508
842
729
655
579
692
827
873
697
623
744
561
839
961
941
599
799
582
532
505
660
773
879
925
738
896
685
660
645
960
683
927
515
912
959
944
977
874
867
705
907
625
715
502
919
842
851
891
847

585
689
positions for insertions
33
12
99
12
54
45
96
93
90
100
43
24
13
77
17
46
40
99
35
3
97
22
97
43
56
23
60
99
14
3
86
73
10
51
47
60
45
43
34
64
62
15
95
58
90
75
8
77
72
47
8
45

1
16
47
83
57
46
25
56
9
60
42
56
38
39
45
10
74
53
97
82
43
13
68
24
94
49
87
45
83
95
61
69
15
51
72
37
16
81
79
47
87
39
30
8
65
78
48
71

6. A Doubly circular Link List of size m consists of unique records where the record consists of numbers consisting of three digits from 500 to 999. Delete n numbers at given position. Count the total number of operations in your program. Print time taken, Memory Used. Also print the size of the list and the list itself at the end of the program. First number gives m and second number gives n.
Input File

200
100
Initial Link list
571
514
565
805
862
649
817
543
641
993
503
759
879
965
973
952
733
564
651
940
926
671
655
663
997
672
927
836
885
662
798
960
892
856
995
945
768
579
523
598
824
559
985
564
976
594
719
962
935
995
634
890

516
949
742
588
598
778
620
705
976
680
789
718
621
601
530
746
972
888
798
771
893
853
881
535
676
816
919
654
977
882
664
704
604
746
784
525
702
953
844
607
903
883
791
930
547
826
656
729
709
634
999
730
748
904
905

506
804
697
736
763
940
848
931
724
560
648
528
953
780
664
899
901
833
788
752
629
908
785
567
505
698
716
574
614
538
773
897
803
874
517
778
882
864
887
958
578
532
570
713
859
888
630
859
650
710
811
752
875
522
912

794
734
903
596
955
833
754
558
563
521
817
598
947
567
877
817
770
864
942
655
799
859
771
759
519
760
796
947
530
844
794
846
791
541
844
903
901
534
positions for deletions
33
12
99
12
54
45
96
93
90
100
43
24
13
77
17
46

40
99
35
3
97
22
97
43
56
23
60
99
14
3
86
73
10
51
47
60
45
43
34
64
62
15
95
58
90
75
8
77
72
47
8
45
1
16
47
83
57
46
25
56
9
60
42
56
38
39
45
10
74
53
97

82
43
13
68
24
94
49
87
45
83
95
61
69
15
51
72
37
16
81
79
47
87
39
30
8
65
78
48
71

7. A Doubly circular Link list of size m consists of records where the record consists of numbers consisting of three digits from 900 to 999. Delete n numbers given. All the occurrences of that number are to be deleted. Count the total number of operations in your program. Print time taken, Memory Used. Also print the size of the list and the list itself at the end of the program. First number gives m and second number gives n.

Input File

200
50

Initial list

927
913
909
998
926
953
909
929
920
976
989
984
981
965
966
978

970
925
930
902
933
952
915
904
932
983
914
916
907
900
969
941
958
915
969
970
937
997
948
971
960
931
911
960
992
948
979
900
950
900
928
937
957
999
937
905
991
920
915
998
932
982
979
927
981
922
953
901
971
934
926

992
991
915
947
958
982
982
997
979
967
990
918
962
902
968
996
966
943
948
928
919
912
974
965
941
913
958
920
992
939
957
981
903
902
923
998
916
987
976
918
919
981
959
934
923
907
930
998
939
987
958
965
996
929
927

960
947
960
981
992
967
996
926
972
936
931
956
939
948
993
943
982
911
999
984
983
948
909
978
947
964
932
993
979
908
968
908
954
964
980
971
936
991
949
927
923
927
921
908
973
928
986
935
918
920
995
946
927
971
983

911
989
974
941
940
915
954
929
940
938
977
992
954
948
917
931
992
947
982
numbers for deletion 996
902
943
916
937
992
979
940
987
909
965
978
999
921
982
969
918
994
913
904
958
948
900
959
992
913
961
957
921
983
995
938
933
967
941
954

936
920
915
966
966
966
914
904
927
940
990
961
920
902

8. A Doubly circular Link list of size m consists of records where the record consists of numbers consisting of three digits from 900 to 999. Insert n numbers again from the same range at given positions. Count the total number of operations in your program. Print time taken, Memory Used. Also print the size of the list and the list itself at the end of the program. First number gives m and second number gives n.

Input File

100

50

Initial Doubly circular link list contents 922

925
960
974
943
916
936
953
981
946
920
910
905
980
924
960
907
973
974
984
960
980
953
991
988
968
964
923
934
927
989
945
946

904
918
975
929
973
965
970
964
956
988
950
914
993
949
925
992
934
991
979
997
939
990
956
959
918
926
939
919
989
978
936
996
904
941
972
999
952
942
937
994
948
926
987
995
922
953
997
946
927
987
952
951
969
913
919

901
907
914
957
912
995
909
903
984
939
964
935
Numbers to be inserted
907
961
937
985
904
999
909
950
919
924
988
998
911
953
974
948
956
942
938
908
926
937
960
968
917
903
908
914
979
904
950
983
973
985
939
905
963
975
905
904
916
997

919
916
958
972
995
902
952
937
Positions for insertion
17
5
9
21
1
7
30
34
32
46
20
7
16
21
8
36
20
14
4
34
26
36
35
25
18
45
11
24
2
6
23
23
10
50
41
22
50
18
26
23
30
44
21
12
39
13

12
12
23
27

9. A stack of size m (Implemented with a linked list) consists of unique records where the record consists of numbers consisting of three digits from 500 to 999. Push n numbers from 1 to 100 at given positions. Also Delete 50 elements from the specified position from the stack. Count the total number of push and pop in your program. Print time taken, Memory Used. Also print the size of the stack and the stack itself at the end of the program. First number gives m and second number gives n.
Input File
100
50
Initial stack elements
971
951
551
590
824
932
669
574
655
985
873
914
855
906
689
751
502
608
815
790
775
532
754
688
650
846
606
763
608
795
890
894
819
577
705
710
560
641
773
811
585

563
700
996
512
541
912
624
796
508
842
729
655
579
692
827
873
697
623
744
561
839
961
941
599
799
582
532
505
660
773
879
925
738
896
685
660
645
960
683
927
515
912
959
944
977
874
867
705
907
625
715
502
919
842
851

891
847
585
689
positions for insertions
67
47
39
47
9
32
49
23
66
41
86
51
22
82
46
79
46
45
7
94
99
46
28
33
97
30
40
28
71
2
44
84
69
92
23
16
26
19
36
27
32
99
17
30
29
89
23
57
43
51

positions for deletion
17
5
9
21
1
7
30
34
32
46
20
7
16
21
8
36
20
14
4
34
26
36
35
25
18
45
11
24
2
6
23
23
10
50
41
22
50
18
26
23
30
44
21
12
39
13
12
12
23
27

10. A Queue of size m (Implemented with a linked list) consists of unique records where the record consists of numbers consisting of three digits from 500 to 999. Insert n given numbers in the queue. Each insertion should be followed by one dequeue. Count the total number of operations in your program. Print time taken,

Memory Used. Print the size of the queue and the queue itself at the end of the program. First number gives m and second number gives n.
Input File
100
100
Initial Queue contents 971
951
551
590
824
932
669
574
655
985
873
914
855
906
689
751
502
608
815
790
775
532
754
688
650
846
606
763
608
795
890
894
819
577
705
710
560
641
773
811
585
563
700
996
512
541
912
624
796
508

842
729
655
579
692
827
873
697
623
744
561
839
961
941
599
799
582
532
505
660
773
879
925
738
896
685
660
645
960
683
927
515
912
959
944
977
874
867
705
907
625
715
502
919
842
851
891
847
585
689

Numbers to be inserted

33
12
99
12

54
45
96
93
90
100
43
24
13
77
17
46
40
99
35
3
97
22
97
43
56
23
60
99
14
3
86
73
10
51
47
60
45
43
34
64
62
15
95
58
90
75
8
77
72
47
8
45
1
16
47
83
57
46
25

56
9
60
42
56
38
39
45
10
74
53
97
82
43
13
68
24
94
49
87
45
83
95
61
69
15
51
72
37
16
81
79
47
87
39
30
8
65
78
48
71

Output File
33
12
99
12
54
45
96
93
90
100
43
24

13
77
17
46
40
99
35
3
97
22
97
43
56
23
60
99
14
3
86
73
10
51
47
60
45
43
34
64
62
15
95
58
90
75
8
77
72
47
8
45
1
16
47
83
57
46
25
56
9
60
42
56
38
39
45

10
74
53
97
82
43
13
68
24
94
49
87
45
83
95
61
69
15
51
72
37
16
81
79
47
87
39
30
8
65
78
48
71

Section-10

Sorting

Input file of five sets to be used for sorting

Set 1: 862650, 206617, 852731, 635836, 172139, 743609, 354281, 862754, 254583, 999744, 609384, 112579, 534871, 489458, 802234, 933761, 975541, 509471, 528390, 554289, 246958, 957749, 58582, 910732, 467417, 20436, 191373, 905371, 238907, 909088, 565190, 80201, 788570, 2832, 761465, 881299, 663990, 962869, 559752, 965036, 533114, 541911, 691455, 900626, 357931, 384399, 17621, 94126, 436162, 94552, 961881, 725238, 388620, 494108, 367337, 485532, 404807, 690355, 44523, 780729, 554265, 995452, 890245, 226672, 673530, 202693, 84446, 93947, 709777, 353094, 492692, 418704, 372412, 705400, 14204, 719190, 902088, 427613, 48295, 427366, 255970, 387423, 988965, 323152, 664303, 105878, 895255, 403971, 77573, 116371, 670955, 442127, 461870, 82038, 71652, 132670, 675537, 292174, 977081, 856290, 254190, 290801, 435640, 663212, 783808, 361127, 115427, 957460, 40509, 698684, 145544, 153378, 639001, 926408, 14886, 578503, 368435, 328683, 807873, 881032, 642742, 840811, 600834, 798117, 445748, 155467, 211442, 252916, 784230, 868672, 498372, 521330, 967314, 355621, 81049, 149264, 493212, 809220, 782917, 665244, 250316, 713578, 596830, 251334, 271061, 111758, 228158, 389712, 262636, 625001, 671183, 813548, 481653, 513245, 941189, 710789, 878995, 664722, 497658, 928491, 177090, 955334, 558483, 843593, 925014, 705018, 292578, 581453, 688088, 786242, 100670, 96144, 264368, 853376, 882880, 355483, 296603, 696845, 656493, 472271, 199117, 73166, 734945, 703956, 658622, 150269, 166247, 575362, 709770, 831361, 131556, 488667, 589082, 866814, 359094, 159529, 904752, 326778, 941668, 904512, 11739, 20334, 327869, 391275, 260528, 895220, 232435, 62354, 835287, 792929, 800940, 303375, 302855, 744544, 401047, 315510, 517319, 218846, 183482, 181276, 194604, 675299, 924233, 738615, 727414, 134836, 827894, 792363, 871648, 59471, 724148, 462740, 919143, 692453, 341586, 447407, 49618, 666396, 420936, 441634, 503830, 270549, 977342, 56613, 696524, 195580, 26289, 642767, 775626, 341103, 337195, 790948, 203673, 190988, 201895, 833612, 917676, 954566, 287957, 397779, 493159, 29736, 372528, 387774, 106856, 158037, 757705, 854986, 946233, 442493, 557043, 804203, 764369, 433092, 393942, 745379, 606861, 927325, 899490, 118058, 212682, 877496, 729467, 616460, 211107, 803878, 664655, 41330, 532407, 310331, 297800, 585669, 678484, 577291, 495189, 448126, 673992, 556765, 754939, 947168, 550997, 160938, 603589, 309286, 14578, 553724, 842373, 975602, 534926, 280106, 74550, 274486, 803399, 471169, 100858, 459368, 222933, 49021, 868016, 151839, 131379, 147042, 252933, 682406, 180094, 704935, 789253, 860001, 166897, 979211, 836092, 318713, 350491, 255641, 5201, 812328, 113276, 699565, 952984, 660855, 459843, 783204, 880628, 951467, 335238, 511574, 64465, 87555, 120776, 135988, 490943, 820925, 995868, 613343, 880727, 710151, 270210, 653082, 294691, 332467, 365371, 736196, 69713, 741582, 7558, 621269, 11848, 83276, 393454, 289975, 685548, 963470, 803620, 118530, 48315, 316863, 491600, 107922, 842409, 295962, 757903, 874646, 653285, 952445, 638056, 793342, 897653, 798702, 881916, 100406, 268745, 357551, 447622, 735307, 353652, 617005, 250467, 182624, 653275, 852915, 719851, 429369, 919544, 393639, 551592, 426758, 902888, 670284, 149511, 48925, 918243, 643892, 387221, 349107, 393578, 128360, 777929, 856103, 663392, 852114, 297888, 751557, 823853, 97645, 322421, 91286, 145092, 357739, 695367, 676026, 380136, 722509, 869638, 82158, 708366, 730673, 834096, 832396, 351254, 81025, 155120, 482842, 94379, 707762, 642065, 919548, 755964, 98173, 655560, 548373, 559579, 598567, 632736, 561201, 677307, 198426, 271408, 494502, 605141, 323807, 986814, 763910, 541239, 633807, 206610, 226701, 932765, 358593, 827535, 521819, 947369, 741519, 655085, 358457, 269850, 187693, 99782, 855823, 807065, 340916, 821758, 415815, 159048, 702220, 585175, 497276, 326416, 893029, 367209, 593515, 261410, 863920, 606804, 475047, 940535, 781864, 398827, 909157, 827637, 194295, 458969, 174327, 30706, 568522, 984136, 734279, 183299, 36154, 619652, 987232, 434153, 752782, 390546, 601018, 584522, 215920, 941487, 746288, 608821, 607582, 347791, 462487, 367456, 880596, 261383, 15510, 8085, 94794, 721565, 817127, 825672, 2617, 634801, 761331, 407562, 600659, 807548, 3009, 661891, 187727, 282131, 466281, 721968, 856810, 844669, 835833, 576627, 369746, 201161, 714135, 143571, 729037, 727455, 693127, 982310, 647211, 146257, 923094, 249568, 949846, 161595, 964352, 894586, 521786, 384265, 237146, 926883, 611815, 68814, 124888, 928596, 171073, 312132, 323929, 263689, 50638, 845786, 524188, 533123, 903394, 868540, 405181, 511964, 162772, 55428, 789525, 694571, 147407, 449792, 667552, 877190, 346993, 457105, 146423, 542405, 796934, 51071, 747548, 383744, 965845, 754885, 416024, 153516, 957197, 746530, 994438, 867298, 340341, 311107, 686815, 641149, 569919, 622146, 5307, 501008, 398374, 816838, 113480, 97303, 641986, 729499, 34535, 25309, 1335, 854538, 452507, 268318, 399062,

637708, 342607, 773303, 826209, 807253, 586765, 529740, 410486, 825300, 201260, 658703, 565825, 30285, 838414, 729851, 494678, 103674, 411367, 243652, 463540, 628890, 554570, 867707, 599791, 380886, 805921, 789485, 598040, 336922, 34108, 870375, 226857, 189465, 927979, 306301, 450939, 103061, 369467, 721323, 127395, 193320, 856613, 375816, 390651, 945440, 392, 77472, 3227, 788752, 68166, 891367, 994867, 39474, 538223, 308023, 305820, 68692, 581023, 779372, 209648, 275653, 646165, 651011, 725014, 414379, 894685, 386308, 794807, 610261, 716737, 296880, 745145, 381028, 88227, 967127, 837857, 936511, 784704, 584116, 817519, 110845, 389398, 2893, 932631, 305460, 85515, 379134, 717184, 521262, 419875, 919789, 855684, 785436, 72830, 929450, 321711, 42777, 839690, 515908, 699974, 381942, 652898, 203417, 175858, 176558, 980605, 171631, 615912, 628075, 265452, 720221, 103491, 93067, 470020, 367247, 682614, 470273, 884654, 762268, 404889, 303491, 294975, 810176, 849890, 489598, 347221, 774863, 752455, 759428, 135730, 486384, 4136, 268193, 916318, 352013, 974948, 948897, 552777, 253515, 333038, 251352, 294947, 65289, 136824, 908565, 353998, 231479, 556960, 617269, 347799, 586465, 564478, 940374, 473165, 669002, 503578, 152287, 117454, 189864, 162067, 95907, 194212, 826770, 25766, 778385, 454142, 685297, 489247, 69351, 118475, 21899, 533667, 165702, 519614, 239942, 26706, 990969, 91123, 916466, 569536, 836577, 329827, 980881, 169808, 538574, 998145, 241621, 302211, 82725, 588366, 220271, 468877, 910982, 559547, 213777, 156044, 433940, 905863, 102131, 490216, 740015, 813065, 985798, 277945, 3634, 246761, 866215, 471424, 214725, 74401, 175114, 621521, 339878, 774736, 756496, 422560, 72253, 385247, 561627, 742033, 108629, 677940, 254168, 740341, 252388, 340229, 148604, 968202, 805845, 728751, 831700, 498804, 858649, 265705, 253271, 919493, 776059, 618766, 174389, 142122, 120238, 923192, 233120, 640933, 394977, 101185, 450241, 514475, 830700, 878921, 731303, 843554, 841094, 109950, 26731, 312458, 143470, 901151, 395867, 48513, 125457, 959866, 654892, 173571, 649661, 205613, 914508, 449498, 242204, 862486, 360758, 788470, 312045, 79898, 415781, 228687, 396053, 323131, 643520, 843035, 759562, 301976, 892979, 88708, 688939, 321233, 375882, 394631, 530316, 226214, 668864, 185294, 49795, 189250, 597629, 720103, 831230, 108044, 760035, 883398, 232284, 602603, 102670, 476674, 429994, 956735, 770530, 210730, 673887, 830470, 66085, 410169, 297842, 897669, 928591, 138216, 78826, 368687, 334200, 455983, 346851, 586816, 790776, 700544, 784121, 667247, 270601, 665716, 745777, 168984, 518414, 207159, 649551, 363193, 995298, 292432, 747542, 99169, 77279, 780476, 436938, 235450, 700482, 201639, 7263, 881510, 759899, 153245, 684832, 679214, 969012, 275673, 710647, 763832, 47609, 778563, 830575, 612451, 287602, 253274, 87641, 174616, 502425, 232289, 746616, 2810, 900023, 129857, 206496, 557199, 436925, 448926

Set 2: 687729, 268130, 333387, 110643, 743239, 396427, 181153, 297913, 656190, 891195, 129296, 496897, 618375, 18330, 920353, 622375, 235136, 398508, 181337, 85733, 708335, 693345, 494641, 1044, 702926, 806914, 716752, 216122, 969714, 576569, 421354, 982909, 868417, 168789, 623764, 987075, 953628, 478371, 936577, 556722, 397816, 636288, 11762, 259150, 947222, 987308, 323201, 232854, 578208, 887292, 736965, 983662, 679314, 118172, 359491, 228308, 148339, 744680, 872411, 859560, 897631, 24480, 867551, 951802, 637818, 288073, 45695, 231369, 959946, 742438, 60830, 726835, 826726, 808770, 26701, 76929, 333411, 557398, 192256, 989008, 227932, 131954, 60213, 567448, 364917, 223412, 284635, 895307, 265886, 446962, 972367, 756306, 256718, 709564, 620840, 646266, 271159, 143101, 650097, 106560, 317616, 269269, 135350, 545002, 342227, 663796, 587640, 500898, 307734, 812033, 967799, 386213, 839858, 953901, 140735, 443129, 127167, 27875, 26325, 969287, 73521, 476757, 414625, 27968, 577400, 53074, 501657, 253116, 797536, 923454, 568275, 639336, 101974, 683985, 43182, 383787, 212696, 115009, 664742, 153457, 361757, 784620, 203387, 659751, 667040, 530864, 868389, 63099, 366499, 84616, 191993, 243571, 301419, 272523, 624294, 218280, 556323, 574019, 95588, 724195, 726715, 130864, 489287, 890278, 276977, 775895, 743674, 772940, 768568, 821412, 370498, 345287, 199378, 588509, 418790, 182804, 418594, 108867, 560318, 31695, 469561, 774227, 907770, 906723, 881768, 302696, 398048, 710039, 958056, 395963, 167500, 425645, 958855, 395818, 427606, 562315, 470320, 673870, 594998, 86663, 569522, 818119, 950233, 297952, 198473, 26235, 209895, 769167, 678051, 232981, 709075, 199591, 797498, 192382, 770721, 7467, 50145, 78182, 141595, 692097, 803851, 479014, 71049, 724574, 351661, 630354, 561273, 30427, 314883, 934015, 46641, 204263, 358285, 347056, 898026, 634901, 286452, 353482, 204977, 344467, 761873, 997546, 721889, 686679, 929137, 458334, 5531, 216170, 401150, 162176, 161830, 394479, 369377, 705332, 146733, 962779, 971977, 383083, 325673, 800933, 301602, 449701, 729068, 458745, 298379, 160309, 246501, 362914, 230547, 907809, 59017, 68083, 683433, 294153, 291593, 366282, 557236, 316541, 665919, 257416, 54974, 708045, 486946, 716804, 884742, 594280, 687248, 412608, 113828,

939090, 811716, 222745, 117919, 245220, 301100, 338333, 50284, 390595, 381380, 613355, 234129, 596524, 872768, 923534, 645425, 889361, 55243, 938048, 355837, 514283, 437935, 943004, 253635, 643092, 685570, 240006, 120934, 984572, 96200, 794106, 408420, 379446, 391988, 91473, 608486, 106028, 75616, 729237, 225889, 475680, 646188, 212259, 810871, 428568, 218986, 554674, 484257, 627707, 391587, 608389, 649132, 622223, 948842, 531862, 172157, 864181, 953995, 690905, 25472, 664735, 126533, 827989, 204129, 64750, 818965, 310879, 694367, 748636, 83997, 95169, 327307, 193894, 395903, 749530, 85955, 568873, 443564, 778823, 699453, 406151, 566808, 368029, 742187, 619803, 430331, 505241, 971116, 138792, 397539, 933866, 451274, 46923, 941647, 217775, 561189, 84234, 54713, 321109, 25420, 399319, 597754, 475673, 763653, 529088, 303263, 139405, 157010, 550091, 76541, 73029, 14040, 291162, 42434, 875630, 82677, 65974, 823775, 854973, 457165, 446738, 141169, 973371, 435996, 957653, 871755, 515032, 785188, 800489, 292254, 822997, 145909, 537396, 385154, 860806, 946782, 958282, 449248, 251583, 319318, 26324, 632446, 879411, 644376, 101670, 837854, 632900, 30026, 294790, 778888, 437420, 147069, 773802, 431525, 724769, 362940, 841903, 875561, 627642, 64927, 269999, 338459, 475042, 885606, 939536, 88622, 825558, 667705, 904378, 385876, 233810, 896688, 40040, 106254, 87908, 441831, 742179, 959545, 231980, 440395, 888998, 380444, 17940, 828446, 360469, 901859, 953958, 889290, 886050, 264038, 288671, 676014, 276525, 418985, 432109, 208841, 749629, 106272, 701166, 201299, 4252, 952887, 915308, 739594, 589060, 389240, 158781, 50015, 534858, 725928, 863630, 754117, 83358, 403959, 376706, 705618, 677234, 664377, 38691, 65958, 183507, 385084, 966372, 650099, 2535, 44798, 484318, 686669, 921843, 447549, 298686, 201764, 372869, 813786, 647008, 540262, 819592, 439702, 238774, 303836, 930283, 270809, 547344, 953914, 130398, 372613, 785005, 56241, 789916, 790957, 732812, 385799, 847771, 184110, 25368, 839549, 549156, 15422, 213119, 757217, 372550, 368010, 600541, 126223, 668386, 929234, 719912, 732399, 521530, 501879, 658571, 254494, 732035, 378829, 818782, 138937, 976571, 43293, 385337, 973276, 85813, 989763, 265940, 512554, 195851, 633799, 950240, 721203, 744661, 85697, 140316, 629518, 940256, 643163, 274999, 486324, 448226, 86075, 988231, 290356, 32446, 231965, 489407, 512530, 193655, 356164, 609444, 282432, 890709, 222121, 402819, 324849, 942449, 33002, 587232, 196211, 107103, 30587, 234179, 747680, 748339, 891079, 500158, 685381, 908434, 127589, 66711, 664126, 439217, 454505, 407203, 426787, 912389, 335744, 132328, 642861, 512109, 72931, 278071, 616570, 278118, 737550, 324293, 278710, 953065, 876182, 501694, 115768, 59904, 246979, 579793, 586621, 632206, 897273, 742944, 278939, 795890, 692632, 852807, 72733, 718062, 460975, 297978, 349598, 479607, 439940, 388212, 647700, 622420, 960780, 778665, 77954, 645669, 69758, 156726, 920098, 18641, 133654, 390699, 475402, 869565, 687238, 704177, 83566, 45170, 324009, 709394, 597773, 486884, 89796, 445616, 716301, 94836, 934384, 306682, 195234, 998441, 376236, 735157, 748304, 243409, 275322, 627799, 860497, 864004, 720987, 951549, 351347, 743776, 993754, 235991, 331283, 69651, 262087, 751399, 407256, 680704, 568044, 569099, 63635, 881054, 851598, 236451, 955453, 902412, 371477, 572209, 239258, 326245, 991782, 496670, 662066, 523332, 316609, 821926, 40351, 228059, 648116, 352273, 663990, 575512, 503116, 445898, 479475, 640622, 732977, 690025, 914236, 108585, 721502, 754995, 25098, 644193, 724729, 500508, 910075, 635718, 457277, 995009, 464983, 730691, 734632, 119218, 269750, 779119, 659295, 69570, 714454, 239667, 340964, 826655, 967990, 284751, 950278, 631194, 235238, 102674, 327815, 766016, 46599, 797379, 298927, 127627, 918605, 797035, 472585, 628055, 997631, 949941, 34696, 72117, 191744, 507878, 451749, 93608, 276599, 994752, 91777, 266317, 245653, 487379, 830290, 802323, 958364, 296907, 793260, 450012, 909225, 67224, 763016, 868322, 648947, 937283, 163775, 89329, 940229, 997983, 371131, 33428, 395260, 226959, 584880, 692120, 96977, 583130, 222030, 76599, 345349, 924203, 961832, 700177, 453601, 358615, 478448, 771312, 137102, 34886, 194709, 8430, 895826, 804464, 54849, 311923, 27868, 215818, 404388, 930501, 639602, 654387, 435225, 585165, 417403, 613420, 535294, 210458, 892698, 829167, 874739, 43730, 279824, 814300, 727850, 681057, 846911, 320871, 687564, 843292, 678935, 954867, 708109, 998128, 714498, 681570, 502017, 573097, 566215, 368379, 827463, 797752, 383943, 806593, 902884, 983527, 256600, 373099, 29254, 715313, 111925, 479183, 148505, 671406, 227368, 203332, 284960, 354619, 853973, 784628, 649858, 196107, 604181, 153296, 342269, 930414, 330119, 70415, 54098, 117011, 466661, 312576, 157902, 134045, 955833, 254978, 9966, 820950, 862874, 39056, 589711, 253804, 492323, 922342, 784838, 324134, 41827, 599482, 591278, 646098, 918081, 663500, 612458, 879383, 440553, 394915, 907498, 382026, 754317, 969611, 511694, 43304, 857699, 881922, 611834, 499444, 640500, 589216, 282644, 594392, 965709, 62765, 169086, 173780, 644432, 655903, 630863, 648132, 389244, 20991, 712976, 396514, 894976, 537286, 599983, 888528, 850940,

875399, 499273, 718517, 944264, 692994, 515619, 92920, 948113, 835975, 99238, 868157, 123340, 587179, 446180, 398571, 759696, 986882, 313590, 348578, 544153, 425386, 701639, 103066, 448123, 35303, 433722, 719102, 635960, 120214, 936820, 143365, 92642, 664894, 605373, 47124, 268987, 320619, 203501, 360899, 10234, 759164, 190174, 411974, 665147, 433808, 451895, 776106
Set 3: 242837, 25145, 27846, 546499, 370158, 823421, 499235, 582832, 942683, 141534, 518489, 436346, 256932, 291928, 730819, 73689, 816825, 9895, 123070, 893392, 210063, 408438, 920502, 76400, 433779, 384486, 547272, 997224, 283718, 275169, 128459, 476714, 603721, 853032, 409347, 135624, 152734, 372238, 872122, 223801, 71969, 772651, 945597, 67774, 842133, 252949, 500713, 108025, 256407, 264037, 228921, 575680, 92369, 269773, 152324, 765180, 201766, 295348, 406970, 287029, 947941, 220163, 41327, 660333, 523032, 60176, 179396, 317688, 782636, 784051, 922782, 810497, 877269, 537084, 232091, 557963, 831852, 238140, 225800, 103774, 33210, 889863, 716060, 579571, 73463, 116712, 733978, 78805, 301372, 815807, 466661, 139012, 930741, 724409, 683937, 727146, 557128, 147353, 611598, 373931, 363424, 909692, 722243, 590481, 555923, 285316, 828047, 209001, 150616, 813801, 11411, 557862, 471827, 35334, 385013, 900862, 846295, 249788, 952584, 952646, 408551, 618523, 85539, 681008, 490496, 706239, 478279, 301225, 283275, 340072, 971932, 140429, 206094, 786829, 759885, 229071, 475997, 869665, 523848, 367999, 577937, 288451, 466996, 673414, 882963, 648986, 613245, 401330, 501375, 844563, 4632, 209041, 301686, 655620, 485697, 921930, 412821, 907435, 216618, 455004, 43845, 753330, 452277, 559265, 108248, 195137, 573794, 81683, 301085, 796250, 179585, 154303, 950549, 651697, 165126, 800929, 859461, 650257, 119209, 658866, 971881, 910002, 868799, 385641, 840628, 444528, 488882, 251541, 915412, 826386, 75883, 802157, 231568, 723696, 667374, 373259, 983352, 882864, 270956, 442610, 265324, 536657, 637162, 636247, 598511, 991910, 354221, 613403, 682344, 959245, 712619, 875368, 855389, 516477, 907136, 195246, 653593, 580194, 393941, 251838, 992979, 286677, 211060, 862601, 261881, 142994, 460758, 343018, 379092, 358505, 829191, 384831, 389545, 513556, 112561, 49061, 134140, 168294, 653129, 349140, 497008, 138223, 217233, 561225, 653914, 561631, 84331, 474486, 515706, 938071, 778749, 848361, 155211, 228776, 907986, 267112, 245216, 27127, 600496, 768131, 652341, 155397, 607206, 271001, 282021, 513203, 269578, 71907, 76778, 138484, 975617, 921222, 948928, 367370, 2668, 931799, 43905, 573995, 439298, 255113, 286526, 934882, 46695, 918280, 826141, 485140, 182583, 945252, 943999, 724174, 212249, 10148, 353428, 717249, 653823, 50847, 779834, 365886, 459355, 277336, 390282, 992756, 921728, 813429, 90997, 241240, 837935, 929312, 501176, 315142, 635439, 32164, 875441, 572898, 790549, 385378, 330907, 746395, 352340, 267942, 88521, 509783, 491759, 24269, 116724, 400606, 187812, 649349, 761045, 743865, 589224, 568874, 255135, 475716, 991198, 220930, 786597, 904087, 856552, 258742, 548395, 904957, 49913, 919356, 469416, 301924, 651404, 29356, 181324, 944913, 251698, 250748, 702896, 226710, 455224, 418400, 30173, 458957, 573786, 29071, 423886, 905427, 741473, 545371, 721726, 972126, 863124, 390305, 290531, 521470, 609765, 691426, 73087, 9145, 157288, 794354, 340749, 195788, 961795, 815806, 484748, 697244, 122170, 304604, 259732, 796642, 179691, 33819, 487660, 935311, 963432, 83901, 75123, 15002, 143873, 563329, 544942, 815868, 168879, 437048, 857316, 812115, 81821, 814606, 759989, 194513, 43075, 109281, 28512, 981683, 358166, 654989, 566360, 154192, 782696, 421193, 857795, 638070, 450618, 630941, 614012, 591425, 307345, 610708, 651592, 588160, 628193, 275547, 924157, 735047, 677947, 741028, 266635, 663482, 161814, 37443, 297194, 401317, 128239, 93407, 494675, 974688, 498657, 527519, 322560, 82589, 504710, 3918, 715969, 723830, 870037, 184482, 55600, 568998, 326509, 812655, 917610, 550459, 794625, 831551, 258165, 679239, 286083, 422275, 477202, 729676, 964544, 969084, 112877, 802156, 567880, 138733, 594124, 107438, 12714, 791548, 974317, 642360, 294483, 778448, 141115, 953437, 261765, 983219, 393501, 462033, 373440, 187033, 402867, 43922, 112300, 626410, 859907, 407419, 333862, 119623, 392728, 984079, 812770, 696708, 283805, 700638, 349259, 113042, 377341, 234871, 360392, 188846, 902301, 885233, 875762, 726797, 480555, 990346, 166706, 890614, 867429, 765326, 944793, 80494, 151047, 876691, 416704, 590341, 978585, 770986, 468909, 352783, 204888, 799847, 830491, 153018, 494969, 272087, 254148, 801746, 973316, 379873, 810950, 236963, 35625, 846065, 226575, 222065, 311711, 335111, 7770, 303576, 303583, 605681, 847222, 442776, 44762, 178622, 490181, 429995, 331821, 865854, 608829, 942408, 629551, 632271, 271445, 463215, 579444, 379728, 730691, 777042, 536026, 980415, 776501, 425577, 800953, 676809, 767982, 668765, 838683, 254033, 529309, 27089, 253211, 576902, 708976, 546249, 123176, 338839, 606449, 86945, 761109, 959657, 614930, 50356, 594534, 392613, 260475, 546954, 769975, 186441, 922912, 223130, 35901, 922266, 866594, 616084, 825135, 241729, 199329, 819069, 937010, 608162, 374585, 183257, 250135, 742629, 384220, 207034,

792929, 547176, 142580, 549373, 739465, 622099, 839311, 204382, 27032, 491517, 94004, 740416, 544227, 169325, 468036, 69346, 247591, 661875, 478147, 741816, 865381, 266623, 220449, 620180, 311513, 454667, 121798, 133062, 2090, 616972, 348641, 611212, 43634, 812771, 872548, 808805, 213110, 86803, 362829, 710244, 170683, 829647, 338299, 737104, 603979, 251732, 439428, 881477, 224124, 314659, 388333, 36091, 629765, 992806, 994042, 805150, 408902, 491163, 591219, 693332, 913651, 661069, 538621, 346502, 149145, 238903, 823415, 88358, 588115, 183785, 101130, 821199, 80733, 761076, 459308, 754950, 898071, 59996, 431068, 311152, 76824, 472462, 558671, 641323, 260047, 688999, 144420, 594801, 838474, 43082, 608840, 624288, 321511, 845827, 364171, 677426, 574984, 684011, 145463, 831903, 43367, 890476, 905991, 74074, 756143, 659413, 484411, 814037, 69744, 234709, 678000, 899881, 660084, 988230, 478374, 869583, 195926, 293116, 261746, 221222, 609116, 922079, 411920, 687526, 341708, 691532, 711892, 948745, 528243, 963514, 60714, 723955, 772250, 359937, 288253, 422478, 876535, 311070, 809177, 327021, 138855, 221678, 965019, 601043, 679125, 479785, 435474, 882883, 926833, 892546, 303627, 62699, 557773, 129418, 985038, 201632, 22425, 373650, 206351, 705748, 417423, 456312, 508769, 282801, 345060, 209184, 390845, 636607, 965351, 857233, 915290, 317115, 969996, 695026, 920845, 576268, 917403, 408404, 397465, 462924, 809021, 942512, 928348, 601090, 442483, 358348, 667938, 63149, 903630, 459143, 993384, 859225, 276654, 239210, 631827, 343070, 120670, 822449, 825143, 412084, 298171, 569766, 447194, 601658, 991754, 389061, 87624, 441136, 956088, 78887, 759200, 675640, 753664, 958293, 784545, 734377, 557005, 348781, 774689, 239823, 878298, 454161, 587298, 779568, 447857, 196934, 218950, 254335, 56552, 194576, 674245, 817540, 160800, 832180, 806584, 125104, 818992, 810751, 651564, 997496, 842927, 399993, 107468, 912287, 520991, 860887, 460095, 763915, 899627, 998296, 346123, 257913, 29387, 27269, 398750, 241359, 552731, 966827, 954689, 458060, 885037, 210156, 67230, 464578, 627211, 254018, 755938, 880239, 872050, 975252, 129373, 333800, 531078, 265234, 962600, 852226, 197820, 656392, 946801, 579168, 68197, 449649, 696991, 701488, 513694, 2677, 864279, 502341, 248176, 237661, 428629, 107888, 235709, 844395, 724423, 976867, 147351, 851989, 845192, 810918, 857203, 289899, 978378, 825196, 915413, 990983, 253893, 183341, 354983, 924362, 16124, 559437, 455920, 627118, 987472, 520198, 517720, 948144, 516734, 822779, 364909, 436925, 79826, 940965, 164200, 4077, 372494, 708181, 654271, 207916, 172087, 378807, 919069, 157009, 9392, 379143, 813051, 857689, 673644, 909669, 855229, 656391, 395706, 616545, 705668, 907143, 21439, 366688, 189237, 233614, 497478, 714195, 547460, 259269, 395893, 269379, 591766, 495210, 260065, 638248, 214520, 675601, 655086, 276497, 673768, 794015, 607227, 201266, 274346, 327754, 215352, 627103, 155092, 180599, 929782, 715142, 739745, 833531, 86858, 217392, 465737, 902819, 239053, 221914

Set 4: 3483, 829353, 970216, 255379, 430254, 113428, 846321, 92568, 750284, 208579, 760454, 299512, 449925, 528355, 734468, 608677, 296038, 184567, 573718, 612378, 201650, 406664, 225843, 608704, 411487, 686622, 783089, 936363, 723789, 858395, 472324, 789839, 636551, 923000, 637659, 836228, 467566, 799428, 267339, 719747, 442730, 597805, 962025, 489638, 609946, 586557, 856816, 585203, 346746, 567987, 390191, 511516, 227794, 162563, 70178, 591138, 98610, 942948, 302339, 515820, 987249, 447927, 268169, 128212, 409623, 212827, 656815, 551109, 658888, 491939, 716722, 416276, 690278, 738606, 624549, 662253, 96653, 884503, 434961, 944247, 330375, 11924, 863029, 147482, 401182, 884441, 534165, 153511, 487372, 473199, 956163, 257708, 88889, 74598, 56531, 256442, 240142, 454928, 469739, 610754, 964101, 818678, 412402, 821092, 296229, 117836, 581488, 226676, 228998, 265655, 975402, 781648, 618128, 469941, 588752, 235026, 189999, 254453, 561468, 248512, 22437, 392519, 17648, 308294, 6069, 629482, 702102, 147666, 444407, 381156, 136733, 132467, 244232, 910418, 536022, 354060, 451986, 680819, 69072, 836641, 714314, 786242, 332536, 713520, 886025, 27132, 864922, 629543, 213539, 315438, 269961, 435913, 972286, 781105, 689968, 142718, 60389, 602022, 930787, 957562, 152345, 882239, 960084, 303195, 589987, 521436, 743014, 219459, 535576, 285146, 439738, 110114, 911890, 308490, 671949, 542605, 520536, 630409, 29914, 789735, 598480, 246941, 486098, 851889, 382761, 992108, 120657, 89167, 358220, 353809, 340133, 758270, 561867, 699739, 959229, 276059, 946594, 503751, 494938, 159249, 505551, 167288, 661950, 857705, 635892, 264058, 199633, 134036, 906414, 48832, 45190, 785016, 218957, 843863, 349735, 623152, 476927, 9955, 830985, 891823, 236981, 685975, 452602, 384941, 786999, 652440, 309791, 930503, 222242, 744465, 488590, 644604, 812634, 141081, 548638, 565291, 32661, 56438, 452065, 762347, 924503, 47885, 898970, 811322, 403530, 197524, 49356, 22072, 279060, 277114, 808999, 831665, 865324, 579867, 199445, 858749, 301764, 991881, 449284, 244110, 443922, 562070, 437186, 377222, 640163, 842992, 461505, 966654, 485703, 470966, 326621, 123496, 418050, 871767, 775042, 648776, 627262, 213471,

787370, 109657, 46842, 97209, 632488, 280822, 551305, 748830, 858846, 70746, 456189, 599477, 1999, 653846, 181017, 55360, 910595, 770300, 729479, 543436, 843935, 355807, 712021, 648053, 687587, 852627, 332400, 38476, 758438, 469423, 314430, 425035, 964173, 288586, 220516, 204103, 950954, 608635, 940516, 517559, 305017, 845567, 700720, 320184, 522444, 585645, 275998, 530876, 217131, 667026, 271597, 986743, 376287, 659688, 463155, 636739, 151403, 214545, 662645, 276196, 163878, 904497, 942894, 562987, 915368, 748807, 805459, 628265, 319133, 767084, 855465, 513598, 892149, 547358, 329007, 623379, 871071, 466962, 515962, 356705, 57438, 954978, 290194, 447735, 102292, 182085, 520486, 311154, 218568, 655650, 494771, 998237, 331096, 848367, 525482, 559283, 45924, 293457, 998744, 500366, 101661, 54107, 991845, 144228, 473430, 733104, 80102, 336728, 318472, 203319, 448665, 207882, 41595, 310673, 344923, 455862, 367254, 243593, 552742, 983139, 548719, 262032, 313680, 76594, 402395, 634998, 77754, 83310, 3872, 844798, 221929, 312655, 38568, 112598, 928518, 688099, 482615, 63996, 667824, 848910, 114604, 311137, 282573, 142575, 261739, 360727, 231925, 64797, 911392, 561420, 591151, 342499, 487109, 801426, 808486, 630149, 263479, 291710, 789165, 749381, 8536, 293706, 977811, 386554, 544903, 616641, 581359, 303328, 938293, 864834, 83303, 467973, 710313, 623711, 493750, 595127, 582528, 915880, 720348, 789746, 597818, 767456, 907186, 260798, 220548, 917658, 145687, 712697, 359139, 112262, 856683, 126595, 496476, 528760, 691818, 703799, 343468, 373918, 768701, 933645, 538842, 951263, 798858, 746149, 611056, 109043, 34668, 60444, 424090, 336309, 713017, 740824, 198991, 678741, 239709, 755585, 42, 10956, 36204, 984109, 887967, 342081, 945646, 795446, 124594, 85381, 504516, 575940, 240277, 439699, 780395, 770114, 886173, 66818, 441036, 687490, 702323, 335601, 719412, 6990, 996043, 865847, 998756, 670805, 873778, 517665, 507930, 105024, 167949, 901301, 454679, 252201, 986873, 182095, 484572, 54300, 611312, 811660, 127304, 171930, 877132, 981312, 205793, 21301, 541505, 825115, 698818, 617491, 411225, 267197, 939797, 963109, 952317, 345103, 981447, 101439, 167631, 578213, 684070, 217876, 281772, 184544, 278864, 162025, 42939, 697288, 850422, 99948, 225169, 419789, 481565, 145307, 704455, 158871, 466183, 773249, 281989, 74167, 173802, 867791, 630938, 737544, 132907, 685545, 353328, 872801, 450969, 327700, 321798, 539022, 371725, 920500, 924084, 622631, 836502, 701535, 659259, 471791, 675996, 204546, 748937, 591503, 256497, 260126, 899088, 343796, 743180, 167071, 245402, 565591, 876198, 724202, 704572, 919726, 426319, 658490, 593150, 234057, 224585, 588902, 370585, 753549, 331911, 868713, 143550, 116789, 449328, 949240, 489977, 929263, 732854, 746533, 842238, 601928, 215264, 751833, 672100, 998455, 777745, 78856, 493322, 335268, 229874, 976113, 239120, 384819, 934356, 28728, 592303, 723786, 310591, 253269, 478658, 404941, 38211, 829675, 861715, 724229, 693798, 793018, 603851, 98393, 179841, 64954, 959999, 591660, 4213, 174937, 822280, 48434, 576432, 736305, 966119, 406758, 53065, 276863, 597948, 459148, 25689, 939195, 636080, 243333, 451221, 748233, 347498, 621661, 737973, 468896, 869463, 635858, 993243, 214971, 597157, 259515, 132891, 181329, 410729, 456436, 747823, 282804, 408506, 187891, 53143, 525987, 423217, 230856, 173527, 414705, 807292, 375656, 889901, 468720, 118755, 220864, 934140, 260588, 675012, 852646, 583821, 931846, 35874, 601994, 868070, 401443, 298208, 315919, 975591, 749536, 112232, 902463, 903261, 328392, 758465, 358362, 654109, 571470, 555561, 882797, 54062, 645521, 653133, 926089, 720001, 256024, 743840, 50714, 18251, 774718, 978651, 890207, 840710, 258232, 321942, 254941, 296387, 242395, 725532, 345142, 74155, 168873, 520253, 575229, 284633, 556767, 768188, 96309, 703314, 904392, 263136, 444674, 743938, 581304, 988699, 262690, 212914, 631113, 495844, 19952, 825635, 783871, 354017, 129147, 532155, 84524, 186435, 150646, 184240, 800347, 82059, 523492, 803068, 857925, 9457, 874387, 122950, 251126, 756384, 971243, 401352, 892336, 903391, 909342, 290508, 737031, 449048, 468726, 922765, 268123, 135548, 86398, 55701, 327946, 113014, 299683, 985054, 814102, 833262, 238369, 133643, 579645, 675737, 2203, 724727, 967103, 780296, 481208, 179933, 222652, 692454, 860286, 441749, 883460, 144784, 681639, 310343, 211827, 323376, 974816, 93480, 168313, 282299, 488559, 959264, 132185, 417401, 638123, 238903, 320590, 502082, 728467, 555362, 130950, 527916, 376115, 95665, 49369, 564728, 290263, 559766, 880903, 784573, 927786, 463082, 407087, 315262, 255588, 102471, 81586, 648577, 488090, 969878, 704416, 321693, 496910, 382234, 9588, 532947, 375593, 170449, 187995, 23415, 615236, 69536, 387637, 572989, 373434, 283574, 96560, 309383, 112644, 648370, 179243, 748122, 610685, 861837, 742902, 299149, 846730, 813496, 258113, 374471, 685655, 999798, 932132, 140358, 631208, 651100, 742858, 758150, 391205, 355326, 692442, 976916, 785433, 246655, 670099, 466818, 189844, 108521, 868344, 455258, 402178, 615163, 185509, 412297, 327427, 374169, 864299, 472115, 122137, 480850, 995845, 147631, 628313, 958670, 971981, 279325, 315446, 86008, 826100, 444438, 117500, 238815, 597198, 353672,

20278, 111583, 707983, 695502, 428835, 531718, 450611, 861305, 80703, 194416, 463635, 919748, 478707, 32676, 463951, 747885, 925611, 674373, 979656, 175486, 705424, 32802, 960712, 298993, 361646, 92373, 979905, 727407, 653238, 270529, 603239, 615070, 48103, 48943, 821587, 262780, 910757, 18702, 475313, 969600, 436251, 847158, 31705, 603810, 431076, 183115, 372898, 247733, 514780, 7137, 878269, 76780, 110227, 771557, 209905, 344968, 560499
Set 5: 787327, 857140, 73328, 500379, 871556, 94029, 799167, 996829, 999688, 582957, 246914, 388836, 973500, 298236, 749788, 171838, 856365, 639135, 971826, 668950, 604, 383971, 902823, 726326, 84151, 887491, 318566, 698743, 602835, 654814, 270943, 734656, 491173, 547414, 874995, 639682, 363711, 695341, 995862, 90698, 901976, 851572, 339002, 640749, 189503, 560287, 656399, 181991, 801392, 632018, 289711, 426854, 763758, 110494, 538411, 730348, 689323, 418888, 515339, 695869, 287273, 478055, 318703, 147731, 643038, 259832, 65270, 922732, 431455, 972889, 537962, 20428, 829929, 547466, 90852, 389848, 992921, 994831, 874771, 96195, 286273, 958390, 140654, 8817, 512117, 478626, 384264, 727098, 902269, 736851, 417840, 380247, 930124, 563969, 134211, 353481, 345509, 578684, 540322, 961271, 810986, 241414, 946332, 756347, 658810, 859770, 655517, 692823, 876760, 325346, 768297, 834816, 113447, 470964, 334906, 52428, 842302, 142208, 388070, 764433, 934608, 960143, 753081, 78591, 641427, 336434, 290054, 253007, 674936, 548197, 730566, 835731, 497272, 776207, 479320, 620293, 50020, 747119, 822603, 480035, 739898, 847311, 579422, 812142, 967015, 247494, 437702, 209467, 123843, 826748, 503647, 948493, 56189, 247354, 215680, 160542, 600769, 209617, 971276, 254294, 79055, 92785, 805368, 485491, 352537, 389203, 24150, 73228, 375399, 86919, 877005, 355829, 413465, 214326, 558959, 895805, 225915, 791405, 50869, 529472, 154318, 675532, 867119, 684104, 24308, 564663, 640814, 537155, 967850, 296644, 607220, 814880, 807645, 559330, 777666, 728871, 382193, 252238, 391697, 125905, 125485, 847797, 638257, 613777, 241730, 818841, 838142, 31794, 782292, 309344, 871826, 814506, 994480, 468244, 887517, 201085, 428960, 968505, 46007, 891214, 271215, 680801, 803162, 197091, 218850, 820346, 885937, 954570, 870105, 942213, 650256, 360126, 409056, 443978, 45555, 77347, 994354, 464626, 971551, 674508, 46041, 781861, 504737, 720609, 873242, 100609, 355598, 534498, 779465, 632701, 481812, 730015, 757864, 873134, 444677, 639663, 761271, 793424, 469412, 130135, 948879, 742374, 103288, 72202, 80869, 859404, 649422, 64337, 144083, 849876, 539352, 821705, 99491, 608701, 504782, 540747, 474793, 187439, 70658, 465774, 600273, 723856, 649347, 436603, 794391, 659623, 595795, 802502, 393368, 264127, 956740, 123402, 814675, 281366, 507699, 486144, 829817, 862066, 527634, 456790, 861038, 803339, 853881, 440470, 509506, 547150, 636345, 458616, 52542, 666889, 266583, 779334, 985240, 319365, 504012, 192710, 267720, 710519, 62298, 122125, 702870, 779027, 73037, 571780, 347535, 473391, 57395, 747909, 874655, 327679, 712230, 378196, 100961, 775456, 707471, 513050, 356960, 138369, 404027, 902035, 321968, 134059, 501344, 663431, 24002, 148122, 319823, 980444, 251240, 38527, 938214, 593907, 963041, 980212, 831223, 66375, 274391, 906308, 931519, 243892, 264646, 437617, 765556, 53437, 688602, 197411, 418927, 305460, 38283, 910954, 803677, 323410, 513297, 593483, 45729, 95755, 877164, 202387, 933143, 102177, 648877, 81579, 277847, 234422, 302377, 24068, 394186, 66661, 536455, 338285, 295122, 853290, 992733, 976013, 700540, 901705, 305693, 898149, 898085, 755455, 406998, 512039, 150294, 824500, 520285, 793542, 862437, 781207, 794255, 675372, 205060, 834462, 786104, 230663, 175000, 809216, 108271, 812375, 11689, 394774, 237491, 743792, 549058, 236917, 624022, 984304, 783114, 341920, 936175, 248985, 808675, 61287, 503629, 608216, 181785, 58300, 664209, 953195, 429626, 121648, 108759, 49317, 676051, 991380, 695048, 591962, 956490, 257272, 937729, 639339, 417553, 222300, 635418, 631277, 426721, 452583, 303175, 631974, 270688, 844420, 571486, 653229, 591883, 129168, 114610, 77328, 453943, 455149, 410509, 762190, 29956, 740037, 844807, 366604, 40988, 575402, 157264, 11745, 873591, 874350, 649386, 305414, 974097, 155286, 778580, 779456, 634132, 953064, 756046, 8285, 665447, 625042, 80696, 221174, 551870, 372020, 53373, 263143, 814098, 249661, 383798, 712977, 664770, 418217, 107942, 300637, 569761, 431409, 491912, 60331, 871639, 734078, 227977, 79949, 47709, 887557, 572677, 762145, 91313, 722700, 976186, 922438, 46257, 382584, 50742, 248199, 156460, 871140, 679179, 456122, 817997, 967059, 22862, 55178, 507781, 569221, 914945, 977097, 164117, 910077, 680766, 547013, 941674, 397160, 796704, 872396, 147536, 321729, 50675, 984194, 160192, 159401, 324018, 145755, 595253, 64976, 146144, 115165, 644305, 982257, 414955, 500322, 785778, 662330, 546889, 431955, 31638, 877403, 686616, 333656, 234435, 584156, 377338, 113812, 560656, 639883, 350963, 14028, 871769, 548483, 874253, 121514, 400325, 732513, 428747, 921105, 545705, 398874, 452262, 746122, 977014, 977978, 454295, 695157, 773542, 769336, 317312, 497262, 383005, 986760, 147354, 322183, 588811, 641168, 616696, 150182, 919785,

501250, 869005, 726506, 351617, 973096, 160395, 257615, 195791, 73731, 722980, 499562, 387505, 512734, 381397, 529439, 283872, 451318, 491772, 424407, 999582, 343721, 912668, 862957, 569431, 474287, 541224, 571837, 467127, 296050, 344365, 245594, 202584, 443905, 793957, 45289, 136203, 350813, 208135, 496029, 177722, 7658, 130565, 335947, 168610, 431266, 316804, 427147, 757148, 862436, 167488, 168840, 352020, 630740, 284008, 840779, 53353, 184172, 19091, 92420, 543292, 823220, 108190, 738232, 376962, 667856, 524775, 916814, 126706, 880478, 153786, 694911, 618608, 203583, 308461, 25881, 163616, 680326, 158421, 777877, 622688, 576738, 824076, 51723, 854840, 148685, 45288, 413124, 321085, 624525, 819114, 279912, 598006, 63469, 555849, 628289, 715916, 763228, 100267, 513164, 740264, 49443, 403997, 650830, 578524, 729324, 328275, 733598, 42968, 600189, 578629, 75673, 463949, 25659, 670043, 831511, 412883, 641815, 357256, 605086, 357347, 293433, 165102, 289093, 709178, 454220, 55451, 243210, 89003, 718307, 697360, 408224, 740619, 513406, 683639, 778334, 713352, 79239, 194504, 194786, 853257, 290404, 6299, 936011, 18982, 311411, 867348, 660816, 152729, 991870, 351522, 910739, 834783, 282863, 133478, 809914, 847440, 236676, 897620, 79565, 712419, 470986, 418521, 451538, 573999, 617600, 785881, 267132, 857365, 741024, 451192, 515760, 384280, 840893, 669264, 842033, 315676, 572052, 627068, 136001, 781998, 923184, 755614, 177075, 41715, 807860, 780939, 332495, 81928, 904006, 736173, 288350, 678280, 466872, 79875, 354025, 787286, 298081, 241432, 646193, 236137, 319234, 804538, 925333, 503809, 188333, 337945, 468633, 771997, 962092, 838652, 415336, 788159, 134913, 856843, 925009, 672835, 957734, 561, 98336, 345029, 811036, 478986, 923960, 144891, 632855, 534118, 121015, 416460, 601385, 442426, 753384, 241124, 567797, 791301, 701470, 379137, 186835, 359167, 72473, 729275, 713040, 477481, 836749, 86475, 201478, 381704, 246058, 226039, 817141, 152706, 323730, 463828, 333583, 317939, 743403, 708925, 844512, 539783, 539234, 43233, 203484, 531919, 110083, 278770, 765116, 489303, 605747, 35618, 374274, 445178, 901302, 994421, 793279, 737761, 406902, 161298, 760591, 162142, 21747, 456629, 339386, 711491, 794249, 571940, 652613, 134435, 176139, 790112, 82108, 706228, 770621, 671431, 739256, 674608, 377346, 30333, 961233, 764108, 613136, 416833, 32228, 510922, 839675, 685969, 511020, 727377, 493324, 453683, 614579, 577922, 781377, 147511, 266975, 248396, 190728, 809581, 116919, 372909, 804481, 894901, 729403, 910719, 846737, 169845, 608873, 294784, 673960, 969245, 62682, 642915, 999667, 207631, 795504, 762266, 449370, 877299, 826397, 474563, 818485, 556865, 966606, 341274, 946110, 677842, 614943, 849682, 32466, 405258, 193583, 550691, 794514, 413089, 313302, 253099, 537563, 453856, 112955, 152069, 801769, 641694, 27743, 686934, 151328, 830551, 173525, 376191, 49664, 799680, 877728, 275590, 14285, 624208, 930808, 760070, 971521, 966155, 526145, 530637, 826162, 634056, 412525, 133098, 426786, 428318, 621156, 110968, 80499, 443166, 479005, 197316, 711239, 251102, 344013

1. a) Take n numbers from x to y (all unique) n=1000 x=1 y=1000000 as given and implement naïve selection sorting. There are five such sets given in the input file. Sort five sets and print the number of swappings, comparisons, memory used, Time taken.

b) Take n numbers from x to y (all unique) n=1000 x=1 y=1000000 as given and implement selection sorting and improve it by taking the min and max element at each pass and swapping it with the first and last elements respectively. Sort five sets and print the number of swapping, comparisons, memory used, Time taken.

c) Take n numbers from x to y (some elements may be duplicate) n=1000 x=1 y=1000000 as given and implement selection sorting and improve it by taking the min and max element at each pass and swapping it with the first and last elements respectively. Sort five sets and print the number of swapping, comparisons, memory used, Time taken.

2 a) Take n numbers from x to y (all unique) n=1000 x=1 y=1000000 as given and implement naïve bubble sorting. Sort five sets and print the number of swapping, comparisons, memory used, Time taken.

b) Take n numbers from x to y (all unique) n=1000 x=1 y=1000000 as given and implement improved bubble sorting program which stops at the pass when there is no exchange in the previous pass and also that the next pass stops at last swap of the previous pass. Sort five sets and print the number of swapping, comparisons, memory used, Time taken.

c) Take n numbers from x to y (some elements may be duplicate) n=1000 x=1 y=1000000 as given and implement improved bubble sorting algorithm which stops at the pass when there is no exchange in the previous pass and also that the next pass stops at last swap of the previous pass. Sort five sets and print the number of swapping, comparisons, memory used, Time taken.

3. a) Take n numbers from x to y (all unique) n=1000 x=1 y=1000000 as given and implement simple naïve Insertion sorting. Sort five sets and print the number of swapping, comparisons, memory used, Time taken.
b) Take n numbers from x to y (all unique) n=1000 x=1 y=1000000 as given and implement Insertion sorting. Implement the improvement of binary search while searching within the subset of sorted elements. Sort five sets and print the number of swapping, comparisons, memory used, Time taken.
c) Take n numbers from x to y (numbers may be duplicate) n=1000 x=1 y=1000000 as given and implement Insertion sorting. Implement the improvement of binary search while searching within the subset of sorted elements. Sort five sets and print the number of swapping, comparisons, memory used, Time taken.

4. a) Take n numbers from x to y (all unique) n=1000 x=1 y=1000000 as given and implement simple Shell sorting. Take the sedgewick sequence to sort the list. Sort five sets and print the number of swapping, comparisons, memory used, Time taken.
b) Take n numbers from x to y (all unique) n=1000 x=1 y=1000000 as given and implement Shell sorting. Take the gap sequence as 513,257,129, 65, 33, 17, 9, 5,3,1 . Sort five sets and print the number of swapping, comparisons, memory used, Time taken.
c) Take n numbers from x to y (numbers may be duplicate) n=1000 x=1 y=1000000 as given and implement shell sorting. Take the sedgewick sequence to sort the list. Sort five sets and print the number of swapping, comparisons, memory used, Time taken.

5. a) Take n numbers from x to y (all unique) n=1000 x=1 y=1000000 as given and implement simple naïve Merge sort. Sort five sets and print the number of swapping, comparisons, memory used, Time taken.
b) Take n numbers from x to y (all unique) n=1000 x=1 y=1000000 as given and implement Merge sorting. Implement the improvement of applying the insertion sort when the list size is less than 16. Sort five sets and print the number of swapping, comparisons, memory used, Time taken.
c) Take n numbers from x to y (numbers may be duplicate) n=1000 x=1 y=1000000 as given and implement Merge sorting. Implement the program without using any additional memory. Sort five sets and print the number of swapping, comparisons, memory used, Time taken.

6 a) Take n numbers from x to y (all unique) n=1000 x=1 y=1000000 as given and implement counting sort. Sort five sets and print the number of swapping, comparisons, memory used, Time taken.
b) Take n numbers from x to y (numbers may be duplicate) n=1000 x=1 y=1000000 as given and implement counting sort. Sort five sets and print the number of swapping, comparisons, memory used, Time taken.
c) Take n numbers from x to y (numbers may be duplicate) n=1000 x=1 y=10 as given and implement counting sort. Sort given set and print the number of swapping, comparisons, memory used, Time taken.
9,3,8,3,7,6,8,7,10,7,3,9,6,10,3,8,10,7,4,6,3,3,6,1,4,7,8,1,5,4,2,3,7,7,1,7,10,2,2,2,7,3,2,5,10,2,8,6,2,6,5,9,7,5,8
,4,3,1,4,5,7,4,5,3,5,10,1,5,1,2,8,7,7,5,8,10,7,7,9,6,3,7,3,10,7,4,5,6,3,5,1,2,8,2,4,10,4,1,1,5,1,2,2,9,7,5,3,10,10
,5,9,6,8,9,6,6,8,6,9,1,1,7,2,8,6,1,3,5,1,10,4,1,6,1,5,8,9,5,7,1,6,2,4,6,5,5,4,3,8,8,1,5,7,9,4,2,3,1,10,2,8,3,10,7,4
,6,8,2,10,3,3,5,7,2,7,8,9,3,7,8,10,7,10,4,9,7,3,9,5,9,8,5,9,4,5,1,6,1,6,5,2,4,1,3,3,3,5,3,5,2,5,2,8,10,7,2,1,7,3,1
4,3,3,6,3,7,8,7,6,1,1,3,8,6,8,9,2,5,4,9,5,3,6,8,6,2,9,5,10,1,5,9,8,5,2,6,7,10,9,10,2,3,6,10,5,4,4,9,8,6,7,2,2,7,5,
5,8,10,3,3,4,3,6,1,6,1,9,7,8,1,6,9,10,5,8,2,2,5,3,10,4,3,6,4,9,6,5,8,7,5,8,3,6,3,6,6,1,9,2,8,10,3,9,4,8,3,8,3,8,1,
8,2,8,10,4,3,7,10,3,8,5,2,2,2,8,9,7,7,5,4,1,9,6,5,3,10,1,1,1,2,10,8,3,9,3,1,7,1,4,3,7,6,9,7,7,9,8,8,7,9,10,7,8,10,
1,3,6,5,7,1,8,4,3,3,9,8,4,5,9,8,6,5,4,9,2,7,1,7,6,9,7,4,5,2,5,3,10,6,3,8,6,6,2,4,4,10,6,3,8,8,3,6,9,5,6,5,3,9,2,4,5
,6,5,6,8,8,6,8,10,5,5,6,9,9,9,8,9,1,1,5,8,7,6,7,2,5,7,6,8,6,5,5,5,3,1,3,9,2,10,3,8,3,1,9,2,5,3,1,7,9,7,4,7,6,9,8,5,
2,2,3,4,9,4,2,7,9,3,10,1,7,5,5,6,2,4,2,5,6,4,6,6,9,7,4,1,7,8,6,10,2,6,1,2,2,10,4,3,9,2,3,10,4,2,2,4,5,3,3,10,8,
6,8,10,4,5,10,3,4,6,7,5,9,8,4,1,10,10,10,6,8,3,8,8,2,4,1,4,6,4,9,8,7,2,2,6,4,3,5,9,7,1,9,8,7,3,8,3,6,4,7,1,5,8,9,8
,2,6,2,10,10,5,7,4,9,2,1,3,3,3,1,7,8,3,4,7,4,10,2,10,7,2,3,8,10,10,5,9,1,10,4,2,8,10,7,10,2,9,10,3,3,10,5,6,4,2,
7,5,2,4,8,10,4,9,8,8,1,1,4,7,1,8,9,1,9,3,3,8,1,7,2,10,2,7,2,5,6,2,1,1,2,9,9,4,5,9,2,9,10,10,6,9,7,2,2,4,2,9,6,7,9,
7,4,9,7,1,4,1,8,6,4,2,8,6,1,2,5,1,5,2,6,9,4,3,5,7,1,5,6,7,5,1,5,3,7,2,1,9,8,3,4,6,6,10,8,7,2,8,6,2,6,10,8,6,1,5,6,3
,7,3,3,10,2,7,9,4,5,4,8,3,6,8,5,7,1,6,3,9,6,5,6,7,5,9,6,9,9,9,5,10,9,6,5,6,1,3,3,10,6,1,9,8,1,2,9,8,3,7,2,10,3,5,6,
10,6,8,6,6,9,9,4,2,9,10,10,8,5,7,6,9,5,3,7,4,9,10,9,7,4,6,7,4,2,1,5,1,7,6,9,2,9,1,7,10,8,3,4,8,8,1,8,9,8,6,5,2,3,6
,3,3,9,8,9,6,8,7,2,1,1,10,3,5,5,4,9,8,1,1,9,1,9,10,3,9,8,1,4,3,8,5,8,4,10,3,4,8,4,1,3,4,7,10,2,1,6,10,5,8,7,3,2,5,
4,6,2,5,9,3,5,7,1,9,3,9,4,4,7,6,1,6,2,6,2,1,6,7,3,4,8,7,2,6,6,7,6,5,7,4,7,6,10,9,10,1,1,5,5,3,3,1,5,10,5,5,10,8,7,
4,9,7,3,9,1,9,3,8,9,9,1,1,3

7 a) Take n numbers from x to y (all unique) n=1000 x=1 y=1000000 as given and implement bucket sort. Use Insertion sort to individually sort the buckets. Sort five sets and print the number of swapping, comparisons, memory used, Time taken.
b) Take n numbers from x to y (numbers may be duplicate) n=1000 x=1 y=1000000 as given and implement simple bucket sort. Sort five sets and print the number of swapping, comparisons, memory used, Time taken.
c) Take n numbers from x to y (numbers may be duplicate) n=1000 x=1 y=10 as given and implement bucket sort. Sort given set and print the number of swapping, comparisons, memory used, Time taken.
9,3,8,3,7,6,8,7,10,7,3,9,6,10,3,8,10,7,4,6,3,3,6,1,4,7,8,1,5,4,2,3,7,7,1,7,10,2,2,2,7,3,2,5,10,2,8,6,2,6,5,9,7,5,8
,4,3,1,4,5,7,4,5,3,5,10,1,5,1,2,8,7,7,5,8,10,7,7,9,6,3,7,3,10,7,4,5,6,3,5,1,2,8,2,4,10,4,1,1,5,1,2,2,9,7,5,3,10,10
,5,9,6,8,9,6,6,8,6,9,1,1,7,2,8,6,1,3,5,1,10,4,1,6,1,5,8,9,5,7,1,6,2,4,6,5,5,4,3,8,8,1,5,7,9,4,2,3,1,10,2,8,3,10,7,4
,6,8,2,10,3,3,5,7,2,7,8,9,3,7,8,10,7,10,4,9,7,3,9,5,9,8,5,9,4,5,1,6,1,6,5,2,4,1,3,3,3,5,3,5,2,5,2,8,10,7,2,1,7,3,1
4,3,3,6,3,7,8,7,6,1,1,3,8,6,8,9,2,5,4,9,5,3,6,8,6,2,9,5,10,1,5,9,8,5,2,6,7,10,9,10,2,3,6,10,5,4,4,9,8,6,7,2,2,7,5,
5,8,10,3,3,4,3,6,1,6,1,9,7,8,1,6,9,10,5,8,2,2,5,3,10,4,3,6,4,9,6,5,8,7,5,8,3,6,3,6,6,1,9,2,8,10,3,9,4,8,3,8,3,8,1,
8,2,8,10,4,3,7,10,3,8,5,2,2,2,8,9,7,7,5,4,1,9,6,5,3,10,1,1,1,2,10,8,3,9,3,1,7,1,4,3,7,6,9,7,7,9,8,8,7,9,10,7,8,10,
1,3,6,5,7,1,8,4,3,3,9,8,4,5,9,8,6,5,4,9,2,7,1,7,6,9,7,4,5,2,5,3,10,6,3,8,6,6,2,4,4,10,6,3,8,8,3,6,9,5,6,5,3,9,2,4,5
,6,5,6,8,8,6,8,10,5,5,6,9,9,9,8,9,1,1,5,8,7,6,7,2,5,7,6,8,6,5,5,5,3,1,3,9,2,10,3,8,3,1,9,2,5,3,1,7,9,7,4,7,6,9,8,5,
2,2,3,4,9,4,2,7,9,3,10,1,7,5,5,6,2,4,2,5,6,4,6,6,9,7,4,1,7,8,6,10,2,6,1,2,2,10,4,3,9,2,3,10,4,2,2,4,5,3,3,10,8,
6,8,10,4,5,10,3,4,6,7,5,9,8,4,1,10,10,10,6,8,3,8,8,2,4,1,4,6,4,9,8,7,2,2,6,4,3,5,9,7,1,9,8,7,3,8,3,6,4,7,1,5,8,9,8
,2,6,2,10,10,5,7,4,9,2,1,3,3,3,1,7,8,3,4,7,4,10,2,10,7,2,3,8,10,10,5,9,1,10,4,2,8,10,7,10,2,9,10,3,3,10,5,6,4,2,
7,5,2,4,8,10,4,9,8,8,1,1,4,7,1,8,9,1,9,3,3,8,1,7,2,10,2,7,2,5,6,2,1,1,2,9,9,4,5,9,2,9,10,10,6,9,7,2,2,4,2,9,6,7,9,
7,4,9,7,1,4,1,8,6,4,2,8,6,1,2,5,1,5,2,6,9,4,3,5,7,1,5,6,7,5,1,5,3,7,2,1,9,8,3,4,6,6,10,8,7,2,8,6,2,6,10,8,6,1,5,6,3
,7,3,3,10,2,7,9,4,5,4,8,3,6,8,5,7,1,6,3,9,6,5,6,7,5,9,6,9,9,9,5,10,9,6,5,6,1,3,3,10,6,1,9,8,1,2,9,8,3,7,2,10,3,5,6,
10,6,8,6,6,9,9,4,2,9,10,10,8,5,7,6,9,5,3,7,4,9,10,9,7,4,6,7,4,2,1,5,1,7,6,9,2,9,1,7,10,8,3,4,8,8,1,8,9,8,6,5,2,3,6
,3,3,9,8,9,6,8,7,2,1,1,10,3,5,5,4,9,8,1,1,9,1,9,10,3,9,8,1,4,3,8,5,8,4,10,3,4,8,4,1,3,4,7,10,2,1,6,10,5,8,7,3,2,5,
4,6,2,5,9,3,5,7,1,9,3,9,4,4,7,6,1,6,2,6,2,1,6,7,3,4,8,7,2,6,6,7,6,5,7,4,7,6,10,9,10,1,1,5,5,3,3,1,5,10,5,5,10,8,7,
4,9,7,3,9,1,9,3,8,9,9,1,1,3

8) Take n numbers from x to y (all unique) n=1000 x=1 y=1000000 as given and implement Radix sort. Sort five sets and print the number of swapping, comparisons, memory used, Time taken.
b) Take n numbers from x to y (numbers may be duplicate) n=1000 x=1 y=1000000 as given and implement radix sort. Implement counting sort to sort individual columns. Sort five sets and print the number of swapping, comparisons, memory used, Time taken.
c) Take n numbers from x to y (all unique) n=1000 x=1 y=1000000 as given and implement Radix sort. Start from the Most Significant Digit first and go towards the least significant digit. Sort five sets and print the number of swapping, comparisons, memory used, Time taken.

9. a) Take n numbers from x to y (all unique) n=1000 x=1 y=1000000 as given and implement Quick sort. Take first number of the list as median. Sort five sets and print the number of swapping, comparisons, memory used, Time taken.
b) Take n numbers from x to y (all unique) n=1000 x=1 y=1000000 as given and implement Quick sort. Take the median of first three numbers as the partitioning element. ort five sets and print the number of swapping, comparisons, memory used, Time taken.
c) Take n numbers from x to y (all unique) n=1000 x=1 y=1000000 as given and implement Randomized Quick sort. Sort five sets and print the number of swapping, comparisons, memory used, Time taken.

10. a) Take n numbers from x to y (all unique) n=1000 x=1 y=1000000 as given and implement max heap sort. Sort five sets and print the number of swapping, comparisons, memory used, Time taken.
b) Take n numbers from x to y (all unique) n=1000 x=1 y=1000000 as given and implement min heap sort. Sort five sets and print the number of swapping, comparisons, memory used, Time taken.

Section-11

Popular problems involving basic algorithmic techniques

1. Write an efficient program to solve Josephus permutation problem which is defined as: Suppose n people are arranged in a circle and we are given a positive integer m <= n. Beginning with a designated first person, we proceed around the circle removing m^{th} person. After each person is removed, counting continues around the circle that remains. This continues until all n people have been removed. The order in which the people are removed from the circle defines (n,m) Josephus permutation. E.g. Josephus (7,3) is 3,6,2,7,5,1,4.
Input file
41,3
Output File
3, 6, 9, 12, 15, 18, 21, 24, 27, 30, 33, 36, 39, 1, 5, 10, 14, 19, 23, 28, 32, 37, 41, 7, 13, 20, 26, 34, 40, 8, 17, 29, 38, 11, 25, 2, 22, 4, 35, 16, 31

2. Write an efficient for checking that whether a given word is an anagram of another word. The word should have same length and same frequency of each letter occurring in the original word. Letters can be repeated with in a word.
Input File
qwertyuiop,oiuytqwerp
Output file
1

3. An array A contains n-1 unique integers in the range [0,n-1] that is there is one number from the range [0,n] that is not in A. Design an O(n) time efficient program for finding that number. You are allowed to use only O(1) additional space besides the array A itself. First number gives n. First Number gives n.
Input File
100
19, 2, 3, 1, 24, 94, 0, 56, 98, 25, 16, 51, 97, 48, 91, 43, 70, 74, 6, 34, 53, 63, 100, 28, 95, 77, 22, 76, 73, 9, 68, 15, 90, 67, 5, 31, 85, 80, 69, 61, 71, 88, 64, 99, 21, 81, 82, 30, 83, 62, 42, 75, 27, 92, 50, 17, 89, 46, 54, 52, 26, 4, 58, 60, 7, 57, 11, 39, 78, 49, 93, 41, 20, 33, 8, 38, 84, 44, 72, 59, 14, 35, 87, 45, 12, 10, 47, 18, 86, 23, 36, 65, 37, 55, 96, 29, 32, 40, 79, 13
Output File
66

4. The first n cells of the array E contains integers sorted in increasing order. The remaining cells all contain some very large integer that we may think of as infinity (name it as maxint). The array may be arbitrarily large and you don't know n. Give an efficient program to find a position of a given integer x (x< maxint) in array in O(log n) time.First number is x.
Input file
465
1,2,3,4,5,6,11,12,13,14,15,16,17,18,19,20,21,22,23,24,28,29,30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50,51,52,53,54,55,56,57,58,59,60,61,62,63,64,65,66,67,68,69,70,71,72,73,74,75,76,77,78,79,80,81,82,83,84,85,86,87,88,89,90,91,92,93,94,95,96,97,98,99,100,101,102,103,104,105,106,107,108,109,110,111,112,113,114,115,116,117,118,119,120,121,123,124,125,126,127,128,129,130,131,132,133,134,135,136,137,138,139,140,141,142,143,144,145,146,147,148,149,150,151,152,153,154,155,156,157,158,159,160,161,162,163,164,165,166,167,168,169,170,171,172,173,174,175,176,177,178,179,180,181,182,183,184,185,186,187,188,189,190,191,192,193,194,195,196,197,198,199,200,201,202,204,205,206,207,208,209,210,211,212,213,214,215,216,217,218,219,220,221,222,223,224,225,226,227,228,229,231,232,233,234,235,236,237,238,239,240,241,242,243,244,245,246,247,248,249,250,251,252,253,254,255,256,257,258,259,260,261,262,263,264,265,266,267,268,269,270,271,272,273,274,275,276,277,278,279,280,281,282,283,284,285,286,287,288,289,290,291,293,294,295,296,297,298,299,300,301,302,303,304,305,306,307,308,309,310,311,312,313,314,315,316,317,318,319,320,321,322,323,324,325,326,327,328,329,330,331,332,333,334,335,336,337,338,339,341,342,343,344,345,346,347,348,349,350,351,352,353,354,355,356,357,358,359,360,361,362,363,364,365,366,367,368,369,370,371,372,373,374,375,376,377,378,379,380,381,382,383,384,385,386,387,388,389,390,391,392,393,394,395,396,397,398,399,400,401,402,403,404,405,406,407,408,409,410,411,412,413,414,415,416,417,418,419,420,421,422,423,424,425,426,427,428,429,430,431,432,433,434,435,436,

437,438,439,440,441,442,443,444,445,446,447,449,450,451,452,453,454,456,457,458,459,460,461,462,463,464,465,466,467,468,469,470,471,472,473,474,475,476,477,478,479,480,481,482,483,484,485,486,487,488,489,490,491,492,493,494,495,496,497,498,499,500,501,502,503,504,505,506,507,508,509,510,511,512,513,514,515,516,517,518,519,520,521,522,523,524,525,526,527,528,529,530,531,532,533,534,535,536,537,538,539,540,541,542,543,544,545,546,547,548,549,550,551,552,553,554,555,556,557,558,559,560,561,562,563,564,565,566,567,568,569,570,571,572,573,574,575,576,577,578,579,580,581,582,583,584,585,586,587,588,589,590,591,592,593,594,595,596,597,598,599,600,601,602,603,604,605,606,607,608,609,610,611,612,613,614,3214678, 3214678

5. Imagine four railroad cars positioned on the input side of the track numbered 1,2,3,4 respectively. Suppose we perform the following operations: move car 1 into stack, move 2 into stack, move 2 out, move 3 into stack, move 4 into stack, move 4 out, move 3 out, move 1 out. As a result the original order has been changed from 1234 to 2431. What permutations of the number 1,2,3,4 are possible that can be get using the above method? Input that permutation and get the output as yes or no indicating it is possible or not. First Number gives number of permutations.

Input File
2
3241,3142
Output File
1,0

6. Each of n boys in an array may have one of the values red, white or blue. Give an efficient program for rearranging the keys so that all red come before white and all white come before blue. It can happen that there are no keys of one or two colours. Operation permitted is swap. First Number gives the total numbers. Don’t use any extra storage.

Input File
100
B,B,B,R,B,W,W,R,W,B,W,B,B,W,B,B,B,R,W,R,B,B,B,B,R,R,B,B,R,B,R,B,R,W,B,R,W,B,W,R,W,R,R,W,
B,R,B,B,R,B,R,B,W,B,R,W,B,B,B,R,B,R,R,W,W,W,B,B,R,W,W,B,R,B,B,W,W,B,W,B,W,B,R,B,B,B,B,B,
R,B,R,R,R,B,B,B,W,W,W,W
Output File
R,R,R,R,R,R,R,R,R,R,R,R,R,R,R,R,R,R,R,R,R,R,R,R,R,R,R,W,W,W,W,W,W,W,W,W,W,W,W,W,W,W,
W,W,W,W,W,W,W,W,W,W,W,B,B,B,B,B,B,B,B,B,B,B,B,B,B,B,B,B,B,B,B,B,B,B,B,B,B,B,B,B,B,B,B,B
,B,B,B,B,B,B,B,B,B,B,B,B,B,B

7. A popular diversion, word find, asks the player to find a word in a square table filled with single letters. A word can read horizontally, vertically or diagonally in any direction. Write a efficient program to find the word.

Input File
ABCDEFGHIJKLMNOPQRSTUVWXYZ

ZABCDEFGHIJKLMNOPQRSTUVWXY
YZABCDEFGHIJKLMNOPQRSTUVWX
LMNOPQRSTUVWXYZABCDEFGHIJK
CDEFGHIJKLMNOPQRSTUVWXYZAB
MNOPQRSTUVWXYZABCDEFGHIJKL
XYZABCDEFGHIJKLMNOPQRSTUVW
DEFGHIJKLMNOPQRSTUVWXYZABC
KLMNOPQRSTUVWXYZABCDEFGHIJ
RSTUVWXYZABCDEFGHIJKLMNOPQ
WXYZABCDEFGHIJKLMNOPQRSTUV
FGHIJKLMNOPQRSTUVWXYZABCDE
EFGHIJKLMNOPQRSTUVWXYZABCD
QRSTUVWXYZABCDEFGHIJKLMNOP
VWXYZABCDEFGHIJKLMNOPQRSTU
JKLMNOPQRSTUVWXYZABCDEFGHI
GHIJKLMNOPQRSTUVQXYZABCDEF
PQRSTUVWXYZABCDEFGHIJKLMNO
UVWXYZABCDEFGHIJKLMNOPQRST
IJKLMNOPQRSTUVWXYZABCDEFGH
OPQRSTUVWXYZABCDEFGHIJKLMN
TUVWXYZABCDEFGHIJKLMNOPQRS
BCDEFGHIJKLMNOPQRSTUVWXYZA
NOPQRSTUVWXYZABCDEFGHIJKLM
HIJKLMNOPQRSTUVWXYZABCDEFG
STUVWXYZABCDEFGHIJKLMNOPQR

WFEQVJ, LKIENSW
Output File
1,0

8. An evil king has a cellar containing n bottles of expensive wine, and his guards have just caught a spy trying to poison the king's wine. Fortunately, the guards caught the spy after he succeeded in poisoning only one bottle. Unfortunately they don't know which one. To make the matter worse the poison the spy used was very deadly. Just one drop diluted even to a billion will still kill someone. Even so, the poison works slowly. It takes a full month for the person to die. Design a scheme that allows the evil king to determine exactly which one of his wine bottles was poisoned in just one month's time while expending at most O(logn) of his taste testers. First integer gives number of bottles and the sequence gives the number of taste testers who died.
Input File
1000
1,3,6
Output File
656

9. In towers of Hanoi problem n disks of different sizes are piled on a peg in order by size, with the largest at the bottom. There are two empty pegs. The problem is to move all the disks to the third peg by moving only one at a time and never placing a disk on top of a smaller one. The second peg may be used for intermediate moves. The usual solution recursively moves all but the last disk from the starting peg to the spare peg, and then moves the remaining disk on the start peg to the destination peg, and then recursively moves all the others from the spare peg to the destination peg. Give the recursive procedure for the above.
Input File
7
Output file
a-b,a-c,b-c,a-b,c-a,c-b,a-b,a-c ,b-c,b-a,c-a,b-c,a-b,a-c,b-c,a-b ,c-a,c-b,a-b,c-a,b-c,b-a,c-a,c-b,a-b,a-c,b-c,a-b,c-a,c-b,a-b,a-c ,b-c,b-a,c-a,b-c,a-b,a-c,b-c,b-a,c-a,c-b,a-b,c-a,b-c,b-a,c-a,b-c ,a-b,a-c,b-c,a-b,c-a,c-b,a-b,a-c,b-

c,b-a,c-a,b-c,a-b,a-c,b-c,a-b ,c-a,c-b,a-b,c-a,b-c,b-a,c-a,c-b,a-b,a-c,b-c,a-b,c-a,c-b,a-b,c-a ,b-c,b-a,c-a,b-c,a-b,a-c,b-c,b-a,c-a,c-b,a-b,c-a,b-c,b-a,c-a,c-b ,a-b,a-c,b-c,a-b,c-a,c-b,a-b,a-c,b-c,b-a,c-a,b-c,a-b,a-c,b-c,a-b ,c-a,c-b,a-b,c-a,b-c,b-a,c-a,c-b,a-b,a-c,b-a,a-b,c-a,c-b,a-b

10. Consider the problem of searching in a sorted matrix. That is, you are given an m×n matrix A, where each entry is an integer. Each row of the matrix is sorted in ascending order, and each column is also sorted in ascending order. Given a value x, the problem is to decide whether x is stored somewhere in the array (i.e., whether there is some i and j such that A[i][j] = x). First number gives m and second gives n.

Input File

8,10

2	4	6	8	12	14	16	18	20	22
3	5	7	9	13	15	17	19	21	23
10	20	30	40	50	60	70	80	90	100
12	22	32	42	52	62	72	82	92	102
13	30	60	70	92	100	139	176	198	1201
26	31	64	74	94	104	144	184	204	1204
76	78	79	80	98	114	154	200	300	1500
100	101	102	103	104	115	165	265	4556	10000

42

105

Output File

3,3

Not present

Section-12

Divide and conquer and Greedy Technique

1. Implement Integer multiplication using Divide and Conquer technique. Assume that digits in the numbers are in 2^n. You are given the input in binary numbers.
Input file
1100001100001111
1010010111110000
Output File
11111100110111110001001000010000 Followed by number of multiplications, Number of recursive calls, Memory used and time taken

2. Implement matrix multiplication using Strassen method.
Input File
First matrix
3 6 9 12
14 35 16 1
6 7 9 13
65 14 58 28
Second Matrix
12 14 19 43
21 31 29 65
3 8 7 2
13 14 81 94
Output File
345 468 1266 1665
964 1423 1474 3003
415 555 1433 1953
1612 2200 4315 6453
Followed by number of multiplications, additions, Number of recursive calls, Memory used and time taken

3. You have n coins that all are supposed to be gold coins of the same weight but you know that one coin is fake and weighs less than the others. You have a balance scale. You can put any number of coins on each side of the scale at one time and it will tell you if the two sides weigh the same, or which side is lighter. Write an efficient program to find out the fake coin in minimum number of weighing. You are given Coin number along with the weights of the coin. You have to identify the Coin with less weight. Standard weight of the coin is 4.
Input File
4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4
4 2
Output File
60
Followed by number of Number of recursive calls, Memory used and time taken

4. Given n points in 2 dimensional spaces, write an efficient program to find a pair of points with the smallest distance between them. It is known as closest pair problem.
Input File
5,9
8,13
5,16
7,20
10,13
1,1
14,19

23,46
29,56
100,100
80,80
60,70
70,60
120,140
111,111
4,9
Output File
5,9
4,9
Followed by number of number of comparisons, number of recursive calls, Memory used and time taken

5. Given n objects, their profits and weights (in this order) and knapsack weight W. Write an efficient program to implement fractional knapsack problem using Greedy Programming. First Number gives n and second number gives W.
Input File
20,100
Output File
10,21
5,120
6,5
18,21
30,10
21,12
45,15
61,24
12,15
89,16
76,61
2,82
34,18
5,11
6,5
7,2
9,1
12,15
27,18
72,100
Output File

Followed by number of multiplications, divisions, time taken and memory used

6. One Processor Scheduling Problem is defined as follows. We are given a set of n jobs. Each job I has a start time t_i and a deadline d_i. A feasible solution is a permutation of the jobs such that when the jobs are performed in that order, then every job is finished before the deadline. Write an efficient greedy program for this which processes the jobs in the order of deadline (the early deadlines processed before the late ones).
Input File
1 2 3 4 5 6
3 2 1 4 3 2
6 8 9 9 14 15
Output File
1,2,3,5,6 Followed by number of comparisons, time taken and memory used

7. You are given starting and ending times of the intervals. Write an efficient program to implement Scheduling of intervals for a particular resource so that there is no overlap in scheduling.
Input File
1 2 3 4 5 6 7 8 9 10 11
1 3 0 5 3 5 6 8 8 2 12
4 5 6 7 8 9 10 11 12 13 14
Output File
1,4,8,11
Followed by number of multiplications, divisions, time taken and memory used

8. Write an efficient program for Optimal merge patterns for merging n sorted lists into a single sorted lists. First number gives n followed by number in each of the list.
Input File
20
6
23
67
12543
321
6578
231
298
980
9981
123921
12
234
657
187
666
290
9
1762
2389
Output File
6,9
12,15
23,27
50,67
117,187
231,234
290,298
304,321
465,625
657,666
688,980
1090, 1323
1666, 1762
2389, 2413
3428, 4802
6578, 8230
9981, 12543
14808, 22524
37332,123921
161253

Followed by number of multiplications, divisions, time taken and memory used

9. Write an efficient program to generate the prefix free codes using Huffman Compression for different alphabets of a given file.

gatcctccatatacaacggtatctccacctcaggtttagatctcaacaacggaaccattgccgacatgagacagttaggtatcgtcgagagttacaagctaaaa
cgagcagtagtcagctctgcatctgaagccgctgaagttctactaagggtggataacatcatccgtgcaagaccaagaaccgccaatagacaacatatgtaa
catatttaggatatacctcgaaaataataaaccgccacactgtcattattataattagaaacagaacgcaaaaattatccactatataattcaaagacgcgaaaaa
aaaagaacaacgcgtcatagaacttttggcaattcgcgtcacaaataaattttggcaacttatgtttcctcttcgagcagtactcgagccctgtctcaagaatgtaa
taatacccatcgtaggtatggttaaagatagcatctccacaacctcaaagctccttgccgagagtcgccctcctttgtcgagtaattttcacttttcatatgagaact
tattttcttattctttactctcacatcctgtagtgattgacactgcaacagccaccatcactagaagaacagaacaattacttaatagaaaaattatatcttcctcgaa
acgatttcctgcttccaacatctacgtatatcaagaagcattcacttaccatgacacagcttcagatttcattattgctgacagctactatatcactactccatctagt
agtggccacgccctatgaggcatatcctatcggaaaacaatacccccccagtggcaagagtcaatgaatcgtttacatttcaaatttccaatgatacctataaatc
gtctgtagacaagacagctcaaataacatacaattgcttcgacttaccgagctggctttcgtttgactctagttctagaacgttctcaggtgaaccttcttctgactt
actatctgatgcgaacaccacgttgtatttcaatgtaatactcgagggtacggactctgccgacagcacgtctttgaacaatacataccaatttgttgttacaaacc
gtccatccatctcgctatcgtcagatttcaatctattggcgttgttaaaaaactatggttatactaacggcaaaaacgctctgaaactagatcctaatgaagtcttca
acgtgacttttgaccgttcaatgttcactaacgaagaatccattgtgtcgtattacggacgttctcagttgtataatgcgccgttacccaattggctgttcttcgattct
ggcgagttgaagtttactgggacggcaccggtgataaactcggcgattgctccagaaacaagctacagttttgtcatcatcgctacagacattgaaggattttct
gccgttgaggtagaattcgaattagtcatcggggctcaccagttaactacctctattcaaaatagtttgataatcaacgttactgacacaggtaacgtttcatatga
cttacctctaaactatgtttatctcgatgacgatcctatttcttctgataaattgggttctataaacttattggatgctccagactgggtggcattagataatgctaccat
ttccgggtctgtcccagatgaattactcggtaagaactccaatcctgccaatttttctgtgtccatttatgatacttatggtgatgtgatttatttcaacttcgaagttgt
ctccacaacggatttgtttgccattagttctcttcccaatattaacgctacaaggggtgaatggttctcctactatttttgccttctcagtttacagactacgtgaata
caaacgtttcattagagtttactaattcaagccaagaccatgactgggtgaaattccaatcatctaatttaacattagctggagaagtgcccaagaatttcgacaa
gctttcattaggtttgaaagcgaaccaaggttcacaatctcaagagctatattttaacatcattggcatggattcaaagataactcactcaaaccacagtgcgaat
gcaacgtccacaagaagttctcaccactccacctcaacaagttcttacacatcttctacttacactgcaaaaatttcttctacctccgctgctgctacttcttctgctc
cagcagcgctgccagcagccaataaaacttcatctcacaataaaaaagcagtagcaattgcgtgcggtgttgctatcccattaggcgttatcctagtagctctc
atttgcttcctaatattctggagacgcagaagggaaaatccagacgatgaaaacttaccgcatgctattagtggacctgatttgaataatcctgcaaataaacca
aatcaagaaaacgctacaccttgaacaaccccttgatgatgatgcttcctcgtacgatgatacttcaatagcaagaagattggctgctttgaacactttgaaatt
ggataaccactctgccactgaatctgatatttccagcgtggatgaaaagagagattctctatcaggtatgaatacatacaatgatcagttccaatcccaaagtaa
agaagaattattagcaaaacccccagtacagcctccagagagcccgttctttgacccacagaataggtcttcttctgtgtatatggatagtgaaccagcagtaa
ataaatcctggcgatatactggcaacctgtcaccagtctctgatattgtcagagacagttacggatcacaaaaaactgttgatacagaaaaacttttcgatttaga
agcaccagagaaggaaaaacgtacgtcaagggatgtcactatgtcttcactggacccttggaacagcaatattagcccttctcccgtaagaaaatcagtaaca
ccatcaccatataacgtaacgaagcatcgtaaccgccacttacaaaatattcaagactctcaaagcggtaaaaacggaatcactcccacaacaatgtcaacttc
atcttctgacgattttgttccggttaaagatggtgaaaatttttgctgggtccatagcatggaaccagacagaagaccaagtaagaaaaggttagtagatttttcaa
ataagagtaatgtcaatgttggtcaagttaaggacattcacggacgcatcccagaaatgctgtgattatacgcaacgatattttgcttaattttatttttcctgttttattt
tttattagtggtttacagataccctatattttatttagtttttatacttagagacatttaattttaattccattcttcaaatttcatttttgcacttaaaacaaagatccaaaaat
gctctcgccctcttcatattgagaatacactccattcaaaattttgtcgtcaccgctgattaatttttcactaaactgatgaataatcaaaggccccacgtcagaacc
gactaaagaagtgagttttatttttaggaggttgaaaaccattattgtctggtaaatttttcatcttcttgacatttaacccagtttgaatccctttcaatttctgcttttttcctc
caaactatcgaccctcctgtttctgtccaacttatgtcctagttccaattcgatcgcattaataactgcttcaaatgttattgtgtcatcgttgactttaggtaatttctcc
aaatgcataatcaaactatttaaggaagatcggaattcgtcgaacacttcagtttccgtaatgatctgatcgtctttatccacatgttgtaattcactaaaatctaaaa
cgtatttttcaatgcataaatcgttcttttttattaataatgcagatggaaaatctgtaaacgtgcgttaatttagaaagaacatccagtataagttcttctatatagtcaa
ttaaagcaggatgcctattaatgggaacgaactgcggcaagttgaatgactggtaagtagtgtagtcgaatgactgaggtgggtatacatttctataaaataaaa
tcaaattaatgtagcattttaagtataccctcagccacttctctacccatctattcataaagctgacgcaacgattactatttttttttttcttcttggatctcagtcgtcgc
aaaaacgtataccttcttttccgacctttttttagctttctggaaaagtttatattagttaaacagggtctagtcttagtgtgaaagctagtggtttcgattgactgata
ttaagaaagtggaaattaaattagtagtgtagacgtatatgcatatgtatttctcgcctgtttatgtttctacgtacttttgatttatagcaaggggaaaagaaatacat
actatttttggtaaaggtgaaagcataatgtaaaagctagaataaaatggacgaaataaagagaggcttagttcatcttttttccaaaaagcacccaatgataata
actaaaatgaaaaggatttgccatctgtcagcaacatcagttgtgtgagcaataataaaatcatcacctccgttgcctttagcgcgtttgtcgtttgtatcttccgta
attttagtcttatcaatgggaatcataaatttttccaatgaattagcaatttcgtccaattcttttgagcttcttcatatttgctttggaattcttcgcacttctttttcccattca
tctctttcttcttccaaagcaacgatccttctacccatttgctcagagttcaaatcggcctctttcagtttatccattgcttccttcagtttggcttcactgtcttctagctg
ttgttctagatcctggtttttcttggtgtagttctcattattagatctcaagttattggagtcttcagccaattgctttgtatcagacaattgactctctaacttctccacttc
actgtcgagttgctcgtttttagcggacaaagatttaatctcgttttcttttttcagtgttagattgctctaattctttgagctgttctctcagctcctcatatttttcttgccat
gactcagattctaattttaagctattcaatttctctttgatc

Output File

t 0

a 10

c 110

g 111 Followed by number of multiplications, divisions, time taken and memory used

10. Suppose we are given an n-element sequence S such that each element in S represents a different vote in an election, where each vote is given an integer representing the ID of the chosen candidate. Without making any assumptions about who is running or even how many candidates are there, design an O(nlogn) time program to see who wins the election S represents, assuming that the candidate with the most votes win.

8,8,4,14,13,10,13,7,13,6,15,9,1,15,9,5,4,13,9,1,6,7,3,11,1,3,12,7,7,14,10,13,1,7,2,13,12,13,7,14,2,10,1,14,15
,11,2,6,2,5,11,13,9,3,2,11,2,13,12,3,5,4,10,13,10,3,15,8,6,7,5,6,13,10,3,6,7,15,13,13,10,4,6,7,7,15,9,9,10,5,7
,2,9,14,1,3,2,13,3,5,3,13,4,7,9,11,12,12,5,3,13,10,10,10,13,3,14,12,6,10,5,10,4,15,12,12,5,2,3,14,10,12,13,8,
2,3,12,6,8,6,4,7,1,10,8,7,13,13,1,11,11,15,2,15,7,8,13,9,3,8,4,7,3,14,2,1,11,8,14,7,11,5,4,5,8,7,2,9,2,6,5,1,8,
10,10,13,6,13,15,5,9,1,15,2,6,3,11,11,3,6,7,12,13,14,14,5,11,8,11,12,13,12,13,10,13,14,4,1,8,10,8,4,14,6,6,1
5,12,10,13,4,2,11,11,4,5,9,1,12,10,12,14,6,8,1,13,15,14,3,13,6,7,6,12,1,9,6,3,2,2,11,5,10,9,6,3,8,13,14,11,14
,12,1,14,8,10,4,4,10,10,5,5,6,3,14,7,15,4,3,4,11,7,2,8,6,10,15,13,2,5,6,9,11,1,15,11,10,10,2,10,5,2,10,7,9,6,1
0,10,1,12,9,11,4,13,3,2,2,2,6,4,8,6,3,4,1,3,14,2,5,1,8,7,4,10,5,1,5,11,14,6,8,2,5,3,6,1,6,15,8,15,1,15,1,12,1,3,
13,1,5,6,15,8,9,12,9,9,3,2,1,5,8,12,13,3,12,2,7,6,7,9,14,4,13,5,4,9,4,12,12,11,5,10,12,10,13,8,7,3,2,14,2,1,2,
1,8,5,10,10,12,5,9,11,6,12,3,1,10,9,4,13,14,8,4,14,2,7,2,3,15,6,8,12,9,13,6,15,2,2,12,13,15,14,13,2,14,6,
12,6,12,2,13,5,11,10,9,7,2,14,1,14,10,2,4,11,15,10,4,8,5,5,14,14,10,4,7,7,8,11,2,5,11,11,11,14,13,4,14,5,6,1
5,4,13,11,5,1,4,2,12,5,10,7,3,7,3,3,5,5,8,8,5,7,8,2,11,11,7,8,9,3,5,15,1,11,8,3,10,9,4,15,8,15,11,10,2,12,2,
6,2,11,2,3,2,15,2,11,14,6,5,4,7,4,2,6,6,13,10,7,4,13,9,10,7,1,8,10,2,9,1,2,10,3,7,2,12,9,3,11,10,5,4,2,15,1,3,2
,3,1,8,7,13,12,2,12,12,4,1,1,9,12,2,14,12,13,9,14,15,10,1,14,15,10,2,1,3,12,13,15,7,9,6,8,4,9,14,15,5,3,5,9,7,
4,6,7,14,1,2,10,4,12,3,3,8,1,14,11,15,15,9,9,15,7,14,4,13,13,10,12,2,6,12,1,12,7,4,14,1,6,9,1,7,4,2,11,4,6,4,
15,9,3,9,2,3,7,3,3,14,5,1,14,12,14,6,13,13,8,9,8,11,1,9,3,7,4,7,10,9,5,13,1,2,1,6,6,12,13,14,4,5,4,2,1,5,5,10,1
,9,5,5,15,13,6,9,11,7,13,6,8,10,11,5,3,8,8,4,11,2,13,9,14,14,8,6,4,9,9,14,5,13,6,12,10,9,1,3,4,14,13,9,8,8,1,
15,2,5,5,6,6,4,7,2,8,6,13,3,4,10,12,4,11,13,12,14,8,3,5,1,8,13,1,2,10,13,7,15,13,9,10,4,5,11,10,12,9,3,1,6,
7,3,5,10,5,7,1,10,11,7,12,8,1,15,11,14,1,2,7,7,8,9,11,12,5,15,4,2,5,2,10,3,11,7,13,7,10,1,3,15,8,12,7,12,2
15,4,9,13,9,15,14,8,14,4,5,7,10,6,12,8,15,10,10,6,5,9,11,11,12,9,11,8,8,14,4,12,8,15,13,4,2,9,9,15,6,10,6,3,1
,10,13,6,11,5,4,10,4,7,2,10,9,6,11,8,4,14,15,13,13,5,13,7,15,13,11,10,7,12,14,1,13,14,4,5,15,10,4,14,15,14,8
,13,3,9,13,11,7,9,8,13,1,8,1,10,13,5,4,3,7,7,12,14,11,10,13,11,14,6,12,4,2,4,9,8,12,13,4,3,15,7,12,6,10,7

Output File

13

Followed by number of number of comparisons, swaps, number of recursive calls, Memory used and time taken

Section 13

Interesting Problems

1. A magic square is an arrangement of the numbers 1 through n^2 in a square array so that the sum of each row and column is the same as well as the sum of the two main diagonals. Write an efficient program to determine whether such a square exists for a given number and if it exists then print the square from m to n.

Input File

3,9

Output File

n=3 sum=15

4	9	2
3	5	7
8	1	6

n=4 sum=34

4	14	15	1
9	7	6	12
5	11	10	8
16	2	3	13

n=5 sum=65

11	24	7	20	3
4	12	25	8	16
17	5	13	21	9
10	18	1	14	22
23	6	19	2	15

n=6 sum=111

6	32	3	34	35	1
7	11	27	28	8	30
19	14	16	15	23	24
18	20	22	21	17	13
25	29	10	9	26	12
36	5	33	4	2	31

n=7 sum=175

22	47	16	41	10	35	4
5	23	48	17	42	11	29
30	6	24	49	18	36	12
13	31	7	25	43	19	37
38	14	32	1	26	44	20
21	39	8	33	2	27	45
46	15	40	9	34	3	28

n=8 sum=260

8	58	59	5	4	62	63	1
49	15	14	52	53	11	10	56
41	23	22	44	45	19	18	48
32	34	35	29	28	38	39	25
40	26	27	37	36	30	31	33
17	47	46	20	21	43	42	24
9	55	54	12	13	51	50	16
64	2	3	61	60	6	7	57

n=9 sum=369

37	78	29	70	21	62	13	54	5
6	38	79	30	71	22	63	14	46
47	7	39	80	31	72	23	55	15
16	48	8	40	81	32	64	24	56
57	17	49	9	41	73	33	65	25
26	58	18	50	1	42	74	34	66

67	27	59	10	51	2	43	75	35
36	68	19	60	11	52	3	44	76
77	28	69	20	61	12	53	4	45

followed by memory used, time taken, number of operations, additions

2. Write an efficient program to print the sum of the numbers occurring in a given alphanumeric string. The numerals coming in continuation should be considered as one number.
Input File
ed34f2gfdt567asw1234a9x3v9
Output file
1858
followed by memory used, time taken, number of comparisons, additions

3. Rotate a one dimensional vector of n elements left by i positions. For instance with n=8 and i=3, the vector abcdefgh is rotated to defghabc. Write an efficient program to rotate the vector in time proportional to n using only a few dozen extra bytes of storage?
Input file
poiuytrewq,5
Output file
trewqpoiuy
followed by memory used, time taken, number of shifts

4. There are n closed doors along a corridor numbered from 1 to n. A person walks through the corridor and opens each door. Another person walks through the corridor and closes every alternate door. Continuing like this, ith person comes and toggles every ith door starting from position i. Determine how many doors are open and which one after n^{th} person has walked through the corridor.
Input file
500
Output file
22
followed by memory used, time taken, number of operations

5. Write an efficient program to find that whether a given string is a rotation of the other given string. For example rotations of car are arc,rca but not rac.
piyush,yushpi
rajul,julra
perminder,rminderpe
charanpreet,peertanrnhc
Output file
1,1,1,0
followed by memory used, time taken, number of shifts

6. Given a matrix, write an efficient program to find the transpose of the matrix. First two numbers give the degree of the matrix.
Input File
2,3
1 2 3
0 -6 7
Output File
1 0
2 -6
3 7
Followed by number of multiplications, divisions, time taken and memory used

7. Given a matrix, write an efficient program to find the Determinant of the matrix. First two numbers give the degree of the matrix.
Input File
5,5
6 4 7 12 1
0 7 4 4 2
9 12 15 41 12
9 7 47 12 4
8 1 4 2 1
Output File
-190747
Followed by number of multiplications, divisions, time taken and memory used

8. Given a matrix, write an efficient program to find the Adjoint of the matrix. First two numbers give the degree of the matrix.
Input File
5,5
6 4 7 12 1
0 7 4 4 2
9 12 15 41 12
9 7 47 12 4
8 1 4 2 1
Output File
-3499 834 1562 2109 -25349
-10716 -33371 5656 2752 -1422
601 2637 822 -4996 4245
-16740 9387 -1631 1095 13158
69784 -2623 -18178 -1830 -29829 Followed by number of multiplications, divisions, time taken and memory used

9. Given Two Lists, write an efficient program to find the Union of two lists where n is the number of elements in each list.Use Arrays for doing operations. First number gives n
Input File
30
7,9,7,9,3,4,1,4,9,15,4,2,6,11,1,6,12,4,4,13,3,10,12,12,12,11,3,11,4,6
17,20,24,25,29,18,25,28,26,18,17,24,16,21,25,22,22,30,15,28,17,25,22,20,25,27,19,24,23,30
Output File
7,9,3,4,1,15,2,6,11,12,13,10,17,20,24,25,29,18,28,26,16,21,22,30,27,19,23
Followed by number of comparisons, time taken and memory used

10. Given Two Lists, write an efficient program to find the Intersection of two lists where n is the number of elements in each list. Use Link Lists for doing operations.
Input File
30
7,9,7,9,3,4,1,4,9,15,4,2,6,11,1,6,12,4,4,13,3,10,12,12,12,11,3,11,4,6
17,20,24,25,29,18,25,28,26,18,17,24,16,21,25,22,22,30,15,28,17,25,22,20,25,27,19,24,23,30
Output File
15
Followed by number of comparisons, time taken and memory used.

Section-14

Trees and Graphs

1. Create a binary search tree of n elements. After creating BST Find the predecessor and successor of given three numbers in the binary search tree. First number gives value of n.
100,532,938,843,327,400,387,8,94,597,313,555,936,498,597,292,306,664,612,664,115,330,783,553,6,654,384,716,190,833,407,190,357,820,222,147,324,455,661,251,109,779,802,541,505,693,687,705,670,517,983,959,605,53,104,224,153,872,39,2,811,763,536,975,822,816,460,782,647,464,32,846,937,172,66,567,739,353,155,458,801,658,998,304,145,615,223,65,10,995,260,209,474,764,543,131,831,658,78,187,591
357,782,65
Output File
353,384,779,783,53,78
Followed by number of comparisons, time taken and memory used.

2. As per Collatz conjecture if n is even, C(n)=n/2 else C(n)=3n+1 then for any choice of n, $C^i(n)=1$ for some i. For example if we start with the number 11 and iteratively compute $C^1(11)$, we get the sequence 11,34,17,52,26,13,40,20,10,5,16,8,4,2,1
Prove the Collatz conjecture for first n numbers
Input File
10000
Output File
contains that in how many terms each number in converges to 1 by Collatz formula
Followed by number of multiplications, divisions time taken and memory used.

3. 1729 is the first smallest number that can be represented as sum of cubes of two numbers in two different ways. e.g. $10^3+9^3=12^3+1^3=1729$
Given a particular number how we can confirm whether it is sum of cubes or not
Input File
1456,2389
Output File
0,0
Followed by number of divisions,multiplications,time taken and memory used.

4. A chocolate bar is modelled as mXn rectangles of m*n pieces. You can take a bar and break it along horizontal or vertical axis into two bars. Write an efficient program to find that how will you break a m*n bar into m*n pieces using as few breaks as possible.
Input File
15,12
Output File
Followed by number of divisions, multiplications, time taken and memory used

5.You are working in the finance office for ABC corporation. There are n employees and each employee received c_i compensation last year and total compensation disbursement was C. This year the company needs to cut payroll expenses to C*. Company wants to put a cap # on salaries such that any employee who was paid more than # last year will be paid # this year. Employees who were paid less than # last year will be paid the same amount as of last year.
e.g. if(c1,c2,c3,c4,c5) = (90,30,100,40,20) and C*=210 then 60 is suitable value for #. Write an efficient algorithm for finding #.Input file first number gives number of employees followed by C* and then followed by the salary of last year of each employee
Input File
14,488,20,100,160,40,76,90,101,209,311,43,24,8,99,36
Output File
40
Followed by number of arithmetic operations, time taken and memory used.

6. Given a list of n numbers. Find Maximum and Nextmaximum numbers from the list. First number gives number of elements in the list.
Input File
100,532,938,843,327,400,387,8,94,597,313,555,936,498,597,292,306,664,612,664,115,330,783,553,6,654,384,716,190,833,407,190,357,820,222,147,324,455,661,251,109,779,802,541,505,693,687,705,670,517,983,959,605,53,104,224,153,872,39,2,811,763,536,975,822,816,460,782,647,464,32,846,937,172,65,567,739,353,155,458,801,658,998,304,145,615,223,65,10,995,260,209,474,764,543,131,831,658,78,187,591
Output File
998,995
Followed by number of comparisons, time taken and memory used.

7. Given a list of n numbers. Find Maximum and minimum numbers from the list. First number gives number of elements in the list.
Input File
100,532,938,843,327,400,387,8,94,597,313,555,936,498,597,292,306,664,612,664,115,330,783,553,6,654,384,716,190,833,407,190,357,820,222,147,324,455,661,251,109,779,802,541,505,693,687,705,670,517,983,959,605,53,104,224,153,872,39,2,811,763,536,975,822,816,460,782,647,464,32,846,937,172,65,567,739,353,155,458,801,658,998,304,145,615,223,65,10,995,260,209,474,764,543,131,831,658,78,187,591
Output File
998,2
Followed by number of comparisons, time taken and memory used.

8. There is a very long stream of integers arriving as an input such that each integer is at most one thousand positions away from its correctly sorted position. Write an efficient program that outputs the integers in the correct sorted order and uses only a constant amount of storage.
640,2540,6810,3673,7639,4004,6024,7912,8506,5723,5385,7627,7385,6944,6650,4358,2768,5449,9464,1226,3173,1280,8057,5096,9043,3004,4515,3104,7839,578,2553,3151,7373,998,21,9033,7726,2108,1403,6948,8338,1674,7359,4861,9077,7884,8760,123,8480,8862,292,7276,3001,5356,2964,2836,2466,7706,9780,9270,2374,259,5055,3209,7237,1505,4307,226,8291,6946,5113,1234,9313,5753,1696,9615,2463,5194,8072,1637,2727,5235,8022,2624,597,6781,7598,7184,1296,6154,3383,2394,6142,4223,6153,8432,4207,9710,8128,9987,9964,7878,6828,9220,9693,3553,5837,4933,7594,2479,6172,8676,7050,458,9368,8809,1765,3339,6253,8979,1951,6510,7780,441,9177,9036,4003,1987,1148,1718,2634,4050,3316,568,3687,1116,6815,2428,2319,3527,5096,4894,7649,870,2919,598,7734,3366,3839,785,6550,8982,6212,7085,3055,3,5506,4018,7239,8032,7022,1119,7945,8879,3105,5626,7087,1175,5754,5651,4516,4655,4847,5278,866,8069,3479,2758,2128,1227,8386,3023,4170,7676,9197,7153,953,4819,8570,88,4434,5857,6550,522,686,8895,7294,3163,4419,2032,6344,9121,1652,9612,9374,9569,1685,2299,9002,4042,6399,5312,1222,5481,9977,5443,3189,9404,3342,7098,8232,24,1053,3044,8098,2160,1183,4570,4809,3350,5809,1293,6746,5476,9941,2996,2932,9301,3522,4710,509,5748,5519,7985,8722,2971,6035,2258,518,2394,7627,9397,4521,3851,958,3581,2658,4468,8368,5964,8478,4262,1185,4162,3660,1224,9813,7906,7245,1369,7865,5430,3541,7558,5952,9674,6147,4398,5592,6927,6371,4974,3938,2512,415,3891,8405,1486,494,1667,506,5988,446,1119,783,4427,7857,3158,9643,7973,3288,4307,3583,5923,7206,4829,3451,2860,7421,3035,5598,8192,5098,9144,5402,6483,7062,4781,5400,7075,8941,3793,7473,7224,3172,1402,2096,887,1420,8438,4689,2853,1039,306,7200,2718,3823,3811,7084,829,1619,3285,1093,9618,4943,9479,3611,1407,3928,1252,8528,7598,3983,8488,2148,8430,1934,788,4632,8313,509,6310,1918,6468,6373,6080,3198,762,2888,3208,1631,8958,591,8412,755,1328,1206,720,5238,3461,3163,7043,1984,6057,5651,6979,7999,920,8916,1330,1249,1578,8666,780,5714,1095,8791,2427,6334,1269,5550,9326,1189,538,3777,905,3482,9511,1968,6649,4179,799,2492,5008,819,6781,6710,7048,5136,2235,2054,2731,1307,2689,6017,7827,2812,5134,7891,9327,5609,2001,4356,8471,1285,9091,4334,2869,8231,3383,9574,2300,5487,593,4728,6385,8671,2225,5796,8426,570,9921,2330,7288,2266,1015,8558,4991,3960,6476,6216,3399,5716,6131,8050,9853,625,3131,6110,6057,1850,7310,5791,7881,9992,1913,1210,8775,3026,3775,7668,4463,5347,4809,9243,760,7559,2872,4743,3054,1700,7570,7552,6301,8720,93,7059,3828,3333,5475,1005,7499,6606,5279,3558,3320,7837,4524,7391,1422,1685,8271,9721,8486,581,9109,6545,9798,9413,7025,4773,9572,4837,2742,1753,8465,3401,4348,9192,7827,8473,3763,1325,3771,9486,7157,7857,4505,5161,8276,6156,8371,8563,401,6815,2633,6055,2170,6968,1946,948,1500,2119,9223,2135,8430,6155,7170,246,4118,2213,7415,7006,9447,1095,2835,8350,6869,4910,4692,3010,3509,9717,8437,2159,6988,6778,2531,1665,2171,5147,6499,9055,8979,2872,4929,4803,4518,7086,9286,4890,6780,51

97,1948,4320,349,6313,2836,6144,3679,5424,2513,4086,3120,5895,4547,1051,7629,5735,6550,4629,6625,5655,4733,3045,3491,1970,5063,449,610,730,5370,4034,9033,8938,6740,8573,2741,1552,6017,3576,859,303,6696,6960,8503,4719,9861,7315,681,9961,1705,3047,3978,5674,6022,6635,5011,2568,1628,1019,2330,219,4679,496,9375,7995,2815,4420,4651,5447,4147,6249,5776,5304,7241,3515,4030,4513,1113,3906,7355,1502,6645,7840,3699,3634,8955,6142,8162,5148,7251,5155,810,2162,1914,3470,3008,3938,4512,6543,7733,6450,8742,5901,6248,8971,4160,5141,852,2500,3864,6994,5545,3299,5003,7942,7452,9148,5161,5396,9738,8310,6745,2218,5043,6916,4755,449,8447,3051,8908,96,3404,2800,4085,8697,5755,5103,1245,8307,6485,5221,4311,5843,7754,522,609,8947,9468,5294,3628,2039,8904,7391,6240,8751,8424,2865,3359,8220,4380,2332,9681,8190,6348,1753,8452,6867,6085,7229,2669,6738,3269,1751,6654,6449,4986,3862,3046,9797,8490,6185,294,5075,2228,644,7626,1562,1442,653,6056,615,8212,9761,3544,8444,2591,3317,4485,1204,7056,8548,4668,8911,591,7283,1479,599,7838,7996,1699,6352,9523,5481,854,7265,7745,9776,4536,4434,9043,4912,9017,7564,7074,7495,6285,7391,8411,4843,1770,9977,9303,6190,4692,1302,991,8057,904,9474,9486,6565,8668,2562,3287,5575,5621,7075,1728,2616,775,4314,4981,9561,8589,6531,517,8896,9492,9270,232,5404,7269,2171,3139,7906,7719,6300,7257,2917,9646,6748,6417,2531,3334,8560,4018,3094,4960,4177,3303,4886,3191,2831,5528,6513,8047,1669,7667,7100,5759,1597,5599,9171,4057,9913,4686,236,6368,7332,5870,292,9714,6637,9230,7148,9341,9023,499,6498,169,1217,3272,2901,2352,2570,9999,5612,5656,5002,6852,5757,1081,6277,9806,3574,3194,4273,8727,6985,417,8427,7359,5867,3746,1106,8700,6145,1933,6925,5372,4617,4880,8023,7431,2857,3987,1502,2327,6642,7796,7660,4816,9816,3803,9767,9047,8645,5397,5165,9242,4387,2198,8682,4545,2169,2142,1542,9528,6709,1009,8313,8513,4696,3307,8529,7019,4840,924,9069,436,6432,8049,6452,5103,672,5870,7006,1056,2289,367,4503,8153,3007,4666,235,707,5261,6642,3886,4388,5373,2015,8347,7544,368,75,1907,8817,4087,2500,9444,3239,7454,5406,5376,2968,4283,2095,6960,39,6659,9511,7748,4013,3270,1587,367,5820,9562,1347,7357,9267,35,2575,2146,1639,8942,4136,4131,4363,8642,251,2298,1316,9990,1125,3684,4537,7712,1892,2489,1639,4548,9545,8303,8708,892,8484,6117,9913,5480,552,8944,7784,7766,209,6066,2453,9599,7518,1770,5746,9838,9346,4704,7404,3086,1581,9591,5004,9107,9512,440,7041,322,5557,7329,547,8129,8770,7012,5928,352,1347,8482,2997,9994,8890,8990,4681,4417,2441,6087,5519,2597,840,793,4643,5700,4086,8396,4525,8624,7623,5232,7686,8968,3401,6997,1946,6539,2115,8116,3340,6906,4931,8893,9831,1453,1429,41,577,911,7669,2004,303,164,5996,6856,7112,7085,9830,1722,2683,6899,9272,7144,4504,6522,7536,2465,1545,5179,303,2010,1919,6489,2823,6658,6009,2938,9473,1421,6200,987,6543,3142,7072,3275,2386,2899,6037,7263,8157,1991,6752,1831,3220,4097,5755,4510,7107,7372,2094,4990,7485,616,6188,4096,9563,6449,4172,5171,4069,8653,8666,4473,878,898,4947,2969,7055,5083,2131,7935,1145,3063,1614

Output File

Followed by number of comparisons, time taken and memory used.

9. Given an array A of ints representing a permutation φ. Update the array A to represent inverse of the given permutation φ^{-1}

Input File

40,100,5,95,95,14,40,3,74,7,41,70,44,74,20,26,51,57,29,35,90,10,63,35,94,100,18,64,14,55,64,65,44,7,51,82,81,11,34,77,100,41,55,57,52,38,45,32,15,11

Output File

Will contain Inverse of the given Input (Because there can be many inverse of the permutation given so specific output is not given)

Followed by number of shifts, time taken and memory used.

10. Implement a function which takes a URL as input and performs the following transformations on it (1) make hostname and protocol lowercase (2)if it ends in index.html or default.html, remove the filename (3) if protocol field is missing add http:// at the beginning and (4) replace consecutive / characters by a single / in the path segment of the URL. N gives number of such sequences

3

www.thapar.edu

http://www.algorithmguru.com/index.html

www.GDEEPAK.com//courses/dnd//default.html

Output file

http://www.thapar.edu

http://www.algorithmguru.com/

http://www.gdeepak.com/courses/dnd/
Followed by number of dominant operations, time taken and memory used.

Section-15
General Problems

1. Given a set S of 25 integers and a CPU that has a special instruction, SORT5 that can sort 5 integers in one cycle. Write an efficient program to identify the 3 largest integers in S using SORT5 and minimize the total number of calls to SORT5.Output file gives total number of calls to function SORT5.
Input File
42,16,60,49,12,39,67,3,47,59,73,96,85,97,43,61,4,58,87,81,74,65,7,55,83
Output File
7
Followed by number of comparisons, time taken and memory used.

2. P is a character String (of length m) consisting of letters and at most one asterisk (*). The * is a wild character. It can match any sequence of zero or more characters e.g. P=sun*day then in "happysundaemonday" there is match of if * is taken as daemon. Write an efficient Program to find a match in a text string T (of n characters). Output file will return 0 is there is no match
Input File
weshouldgivesomeofourtimetothenation
gi*tot
manypeoplehassacrificedforourcountry
has*nuoc
Output File
givesomeofourtimetot
0
Followed by number of comparisons, time taken and memory used.

3. Write an efficient program to implement Boyer-Moore algorithm for string comparison.Output file gives the starting position in the string where the pattern is matching, if any.
Input File
puszmmwetjsghsfznrajjkltgbopmblurtkqzfrprsugdfsdlnfwnzwtdatjsxgmqlkmruqehtveajhwikyekmawpyhplxfxqsrb
rsugdfsdlnfwnzwtdatj
Output File
40
Followed by number of comparisons, time taken and memory used.

4. Write an efficient program to implement Knuth Morris Pratt Algorithm for string comparison. Output file gives the starting position in the string where the pattern is matching, if any.
Input File
puszmmwetjsghsfznrajjkltgbopmblurtkqzfrprsugdfsdlnfwnzwtdatjsxgmqlkmruqehtveajhwikyekmawpyhplxfxqsrb
rsugdfsdlnfwnzwtdatj
Output File
40
Followed by number of comparisons, time taken and memory used.

5. Write an efficient algorithm to implement Binary Search.First number gives n.
Input File
500
1, 5, 6, 11, 20, 21, 22, 26, 27, 28, 31, 36, 39, 42, 44, 45, 47, 48, 49, 51, 53, 57, 63, 66, 69, 73, 77, 84, 93, 94, 95, 99, 108, 112, 113, 115, 117, 120, 124, 126, 127, 130, 131, 134, 135, 142, 143, 144, 145, 150, 152, 159, 163, 166, 167, 170, 172, 176, 182, 185, 188, 190, 191, 192, 194, 195, 196, 197, 198, 199, 203, 204, 206, 209, 210, 218, 220, 221, 227, 228, 232, 239, 240, 245, 252, 254, 256, 261, 265, 268, 272, 282, 284, 291, 292, 299, 300, 303, 305, 309, 311, 315, 320, 322, 336, 344, 347, 353, 354, 358, 359, 364, 366, 368, 369, 370, 373, 375, 379, 381, 386, 388, 389, 405, 414, 420, 421, 432, 435, 437, 445, 456, 459, 472, 476, 484, 486, 490, 492, 497, 500, 508, 510, 516, 517, 518, 520, 529, 532, 534, 536, 543, 546, 555, 559, 560, 561, 567, 574, 575, 577, 580, 587, 588, 595, 596, 598, 603, 608, 614, 615, 624, 625, 631, 637, 638, 639, 659,

662, 665, 667, 670, 671, 673, 679, 683, 691, 692, 694, 701, 702, 704, 710, 711, 715, 720, 731, 734, 736, 752, 754, 760, 775, 777, 786, 788, 790, 804, 810, 814, 816, 822, 830, 835, 837, 848, 852, 856, 858, 863, 869, 872, 878, 882, 883, 887, 889, 894, 899, 905, 907, 918, 922, 929, 931, 934, 936, 943, 962, 965, 973, 976, 977, 983, 984, 986, 989, 992, 994, 1000, 1007, 1008, 1011, 1012, 1015, 1018, 1023, 1025, 1027, 1029, 1039, 1044, 1047, 1048, 1052, 1059, 1073, 1074, 1077, 1078, 1082, 1083, 1095, 1103, 1104, 1106, 1108, 1113, 1115, 1125, 1127, 1128, 1134, 1135, 1137, 1138, 1142, 1146, 1151, 1155, 1157, 1158, 1160, 1165, 1167, 1177, 1182, 1186, 1188, 1199, 1200, 1202, 1216, 1223, 1227, 1238, 1242, 1246, 1248, 1258, 1259, 1261, 1262, 1267, 1271, 1272, 1284, 1286, 1288, 1291, 1292, 1303, 1304, 1307, 1308, 1310, 1317, 1327, 1329, 1331, 1335, 1338, 1343, 1346, 1350, 1354, 1356, 1357, 1359, 1367, 1371, 1372, 1373, 1374, 1375, 1376, 1381, 1388, 1393, 1395, 1410, 1414, 1419, 1421, 1435, 1437, 1439, 1443, 1446, 1447, 1450, 1452, 1454, 1455, 1457, 1458, 1459, 1473, 1476, 1478, 1479, 1480, 1485, 1487, 1493, 1496, 1503, 1510, 1516, 1524, 1526, 1529, 1534, 1536, 1537, 1538, 1544, 1547, 1548, 1557, 1569, 1573, 1575, 1577, 1580, 1582, 1584, 1585, 1586, 1588, 1601, 1602, 1608, 1610, 1612, 1614, 1616, 1619, 1621, 1626, 1631, 1633, 1635, 1636, 1642, 1648, 1651, 1656, 1660, 1663, 1669, 1671, 1673, 1676, 1681, 1686, 1687, 1692, 1696, 1700, 1718, 1721, 1724, 1727, 1730, 1735, 1748, 1759, 1760, 1765, 1773, 1774, 1775, 1776, 1777, 1784, 1794, 1798, 1804, 1809, 1813, 1818, 1819, 1824, 1825, 1827, 1830, 1835, 1839, 1841, 1851, 1852, 1854, 1855, 1856, 1857, 1862, 1863, 1866, 1877, 1887, 1888, 1889, 1898, 1899, 1902, 1904, 1908, 1913, 1919, 1924, 1928, 1930, 1931, 1940, 1943, 1949, 1952, 1959, 1969, 1973, 1978, 1979, 1980, 1983, 1987, 1994, 1996, 1998, 1999
21,1998
Output File
5,498
Followed by number of comparisons, time taken and memory used.

6. Write an efficient program to find the square root of given n numbers without using square root function. First number in the input file gives n.
Input File
5,10,100,343,2657,10023
Output File
3,10,18,51,100
Followed by number of comparisons, time taken and memory used.

7. There is a list of distinct sorted numbers stored in array A. Design an efficient algorithm for finding an index i such that A[i]=i or concluding that no such index exists. First number gives the total number of elements in the list.
Input File
30,2, 3, 4, 6, 7, 9, 24, 29, 32, 34, 37, 38, 43, 44, 45, 48, 65, 69, 75, 76, 79, 80, 81, 82, 86, 90, 93, 96, 98, 99
Output File
Does not exists
Followed by number of comparisons, time taken and memory used.

8. Find the index of the first occurrence of an element k in the sorted array A whose length is unknown in advance, however accessing the array beyond its limits will throw an exception. Output should be -1 if element is not present in the array.
Input File
86, 120, 217, 338, 424, 468, 546, 576, 614, 670, 702, 705, 777, 812, 912, 1006, 1007, 1036, 1078, 1079, 1085, 1103, 1138, 1165, 1171, 1177, 1219, 1250, 1467, 1472, 1538, 1559, 1572, 1611, 1624, 1628, 1632, 1699, 1756, 1758, 1764, 1802, 1860, 1923, 1925, 1932, 2065, 2115, 2146, 2181, 2229, 2264, 2308, 2328, 2398, 2409, 2431, 2441, 2512, 2517, 2522, 2539, 2552, 2559, 2584, 2714, 2727, 2747, 2780, 2789, 2794, 2805, 2829, 2853, 2892, 2913, 2919, 2930, 2943, 2947, 2983, 2992, 2996, 3007, 3032, 3071, 3073, 3078, 3081, 3128, 3134, 3213, 3238, 3243, 3265, 3297, 3309, 3402, 3466, 3475, 3492, 3504, 3529, 3532, 3574, 3604, 3606, 3648, 3653, 3666, 3687, 3707, 3784, 3808, 3820, 3883, 3892, 3919, 4025, 4045, 4068, 4127, 4162, 4186, 4187, 4193, 4247, 4277, 4386, 4408, 4440, 4470, 4501, 4642, 4666, 4688, 4707, 4793, 4825, 4878, 4925, 4954, 4991, 5028, 5030, 5046, 5140, 5149, 5178, 5229, 5243, 5313, 5387, 5399, 5451, 5458, 5460, 5488, 5494, 5498, 5524, 5555, 5616, 5644, 5668, 5702, 5737, 5742, 5784, 5794, 5800, 5900, 5904,

5932, 5955, 5957, 5966, 5991, 6054, 6095, 6196, 6219, 6265, 6288, 6306, 6310, 6311, 6315, 6320, 6374, 6384, 6451, 6470, 6523, 6541, 6546, 6555, 6562, 6607, 6612, 6621, 6636, 6648, 6782, 6840, 6861, 6884, 6896, 6933, 6981, 7030, 7062, 7134, 7180, 7222, 7249, 7328, 7366, 7368, 7378, 7385, 7437, 7448, 7461, 7473, 7508, 7565, 7591, 7600, 7621, 7626, 7636, 7755, 7799, 7805, 7815, 7927, 7967, 7994, 8005, 8016, 8020, 8047, 8109, 8163, 8191, 8235, 8251, 8271, 8287, 8288, 8298, 8314, 8329, 8333, 8437, 8456, 8518, 8562, 8700, 8739, 8788, 8797, 8837, 8845, 8869, 8889, 8907, 8945, 8951, 8980, 9063, 9095, 9099, 9109, 9127, 9166, 9224, 9282, 9291, 9449, 9464, 9477, 9493, 9514, 9537, 9554, 9564, 9582, 9628, 9631, 9723, 9791, 9799, 9866, 9887, 9943, 9947, 9957, 9974
2146,9974
Output File
49,300
Followed by number of comparisons, time taken and memory used.

9.Given a dictionary of English words, return the set of all words grouped into subset of words that are anagram of each other.
Input File
jpkr,bmlp,nwcj,gtjb,qpvj,cdkg,xlca,ldcm,vooh,jlcn,udsw,pxcq,xmms,kieu,rakj,armv,ukba,smvb,wash,kxvd,vrzd,nrxg,dtri,scix,dfac,ohmn,tzdz,plio,dlpe,qxzo,npqy,cuxp,mffv,fijg,ojor,zpfx,oyom,hutw,wrww,pgmp,uyqa,ixsh,foyo,meoi,bwmr,bwhh,hzma,vznu,yznr,rdug,yyjy,dbhg,pwhe,qbjo,aqmw,zkhh,rdar,nqsy,wjnf,pdpq,ubnn,dgon,gweh,ungc,epxa,nzbb,ught,awow,qylj,itug,tmyx,krlo,gikv,ntaq,stne,pcbj,bssm,oypc,cyfc,dzid,ffui,uzmu,btkn,ngrj,imao,hsys,defl,ypsn,spqn,wjhy,fywg,myhu,ilfp,zlxk,jejl,ktit,vouf,jonp,ojal,zflm
Output File

Followed by number of comparisons, time taken and memory used.

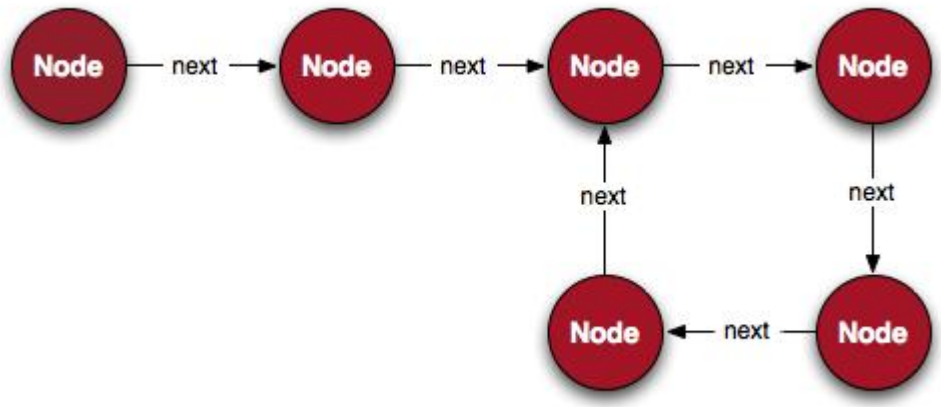
10. You are given two sorted arrays of length m and n. Give a O(log m+ log n) time algorithm for computing the kth smallest element in the union of the two arrays. Keep in mind that the elements may be repeated. First two numbers gives the size of the arrays followed by four values of k.
Input File
100,100,20,109,30,120
4, 5, 6, 7, 11, 13, 14, 18, 19, 21, 22, 24, 26, 29, 30, 32, 33, 38, 39, 40, 41, 43, 46, 47, 48, 50, 55, 58, 59, 61, 62, 63, 64, 69, 71, 72, 73, 76, 79, 81, 83, 85, 89, 90, 93, 96, 99, 100, 104, 105, 106, 109, 115, 116, 122, 126, 127, 128, 130, 131, 132, 133, 136, 138, 139, 141, 142, 143, 145, 146, 148, 149, 150, 151, 153, 154, 157, 158, 159, 161, 162, 166, 168, 170, 171, 172, 173, 174, 175, 176, 180, 185, 186, 187, 188, 190, 191, 192, 194, 199
9, 12, 13, 14, 17, 22, 24, 29, 31, 42, 44, 50, 53, 65, 66, 73, 80, 82, 84, 86, 96, 103, 106, 107, 115, 124, 126, 134, 135, 139, 144, 154, 163, 165, 169, 174, 177, 182, 189, 198, 200, 202, 206, 210, 212, 220, 226, 227, 230, 231, 234, 244, 245, 247, 259, 277, 281, 282, 284, 286, 294, 295, 301, 304, 305, 310, 313, 316, 318, 330, 335, 340, 344, 345, 358, 359, 368, 369, 373, 377, 386, 388, 392, 394, 399, 406, 422, 427, 431, 434, 436, 447, 449, 453, 461, 469, 474, 475, 495, 498
Output File
26,495,41,434
Followed by number of comparisons, time taken and memory used.

Section-16

1. Write an efficient program for random number generation between a given range of numbers. Input file has n different ranges. First number gives n.
Input File
6
1,30
20,60
100,200
12,45
9,82
500,800
Output File
Random numbers in the given range
Followed by number of operations, time taken and memory used.

2. Write an efficient algorithm for random series generation between a given range of numbers. Input file gives the count of numbers that are to be generated between this given range.
Input File
10, 20,100
100, 50, 100
200,100, 10000
Output File
It will contain 10,100 and 200 random numbers respectively within the given range.
Followed by number of operations, time taken and memory used.

3. A node in a linked list might point to a previous element in the list. This is bad for many reasons, not the least of which is that any loop which iterates over all the nodes in the list by accessing the next node will never terminate. So, it becomes important to detect when linked lists have loops. Here's what one such errant linked list looks like.



Input File
A link list with 100 nodes
Output File
returns 0 if there is a loop with the position where there is a loop
returns 1 if there is no loop
Followed by number of comparisons, time taken and memory used.

4. Write an efficient program to Implement a stack using two queues which means implement push and pop function on two queue structures.
Input File
push 5,push 8,push 18,pop,push 24,push 34,push 87,pop,pop,push 98,push 182,push 109,pop,pop,pop,push 300,push 121,push 111,pop,pop,pop
Output File

24, 8, 5
Followed by number of enq,deq,push,pop, time taken and memory used.

5. Write an efficient program to Implement a queue using two stacks by implementing enqueue and dequeue operation on two stacks.
enq 5,enq 8,enq 18,deq,enq 24,enq 34,enq 87,deq,deq,enq 98,enq 182,enq 109,deq,deq,deq,enq 300,enq 121,enq 111,deq,deq,deq
Output File
300,121,111
followed by number of enq,deq,push,pop, time taken and memory used.

6. Write an efficient program to find all the substrings that are being repeated in a given string. For example in the string "hellohowareyouhel" substring 'hel' is repeated. Don't consider single alphabet repetitions.
Input File
anynationstrivesonitsculturalheritageandshouldbeproudofitsnationalcharacter
Output File
nation
Followed by number of comparisons, time taken and memory used.

7. Write an efficient program to implement AVL Trees with insert and Delete functions.
Input File
Numbers to be inserted

64	774	550	598	720	962	175	195	278	308
831	35	128	833	92	229	504	671	859	481
112	983	175	152	644	575	868	272	476	793
397	462	159	508	626	348	663	811	744	440
601	855	611	589	167	728	378	815	533	32
192	212	389	288	415	460	932	794	854	623
388	772	7	773	969	245	770	946	883	903
783	838	851	440	862	195	331	713	436	123
138	894	416	642	10	926	755	60	143	894
496	942	5	916	851	628	828	210	639	548

Numbers to be deleted

474	821	694	321	916	19	918	727	596	105
691	863	946	863	878	304	175	743	213	506
241	641	340	410	845	613	351	530	345	277
91	878	592	586	287	514	795	464	833	801
815	76	783	239	767	318	87	440	210	471
533	110	989	844	601	951	529	628	672	780
442	705	554	375	270	701	316	759	812	989
42	716	961	736	889	91	595	934	656	817
298	67	963	888	382	13	780	424	442	999
505	942	407	540	203	983	25	535	518	173

8. Write a function that takes a set of open intervals on the real line (a_i,b_i) for $i \in \{0,1,\ldots,n-1\}$ and determine if there exists some interval (a_l,b_l) that is completely contained inside another interval (a_m,b_m). If such pairs of intervals exist, then return one such pair.
Input File
1,20,2,34,3,23,5,9,12,23,21,45,26,76,54,13,24,45
Output File

Followed by number of comparisons, time taken and memory used.

9. You are given a number of identical balls and a building with N floors. You know that there is an integer X<N such that the ball will break if it is dropped from any floor X or higher but will remain intact if

dropped from a floor below X.Given K balls and N floors what is the minimum number of ball drops that are required to determine X in the worst case.Input is in the order (K,X,N)
Input File
20,10,50
Output File

Followed by number of comparisons, time taken and memory used.

10. Consider an mXn array A containing integer elements (positive, integer and zero). Assume that elements in each row of A are in strictly increasing order and the elements in each column of A are in strictly decreasing order. Write an efficient program that counts the number of zeros in A. First number gives m and second gives n
Input File
8,8

1	5	8	90	120	140	160	1700
0	1	3	23	24	28	98	900
-3	0	2	13	16	18	40	567
-9	-2	1	10	12	14	20	345
-10	-8	0	6	8	10	15	234
-34	-23	-10	4	5	7	10	123
-98	-30	-20	0	3	5	9	12
-908	-300	-100	-10	0	2	8	10

Output File
5
Followed by number of comparisons, time taken and memory used.