1	Explain All the phases of a compiler in all the details with a suitable example. Make Charts, Tables and Diagrams wherever required.
2	1. Convert the following program into machine language. The Format should be like Address Instruction For example the first line will be 484 BALR 2,0 SAE START 484 ARCHON EQU 1 DEPUTY EQU 2 TREAS EQU 3 BALR 2,0 USING *+ARCHON-DEPUTY-TREAS,TREAS LM 1,6,POINT USING BETA, ARCHON, DEPUTY EXTRN BACK ENTRY BEACON BEACON CR DEPUTY,ARCHON BNH POINT+SAE-BEACON LA 7, =A(BACK) CLI HOLE, X'90' BR 6 DC H'64',X'40',B'1000000',C' ' DROP DEPUTY DC 15X'0',6F'180' BETA MVC POINT+20(4), =H'43' //LENGTH 6 MVC POINT+16, =H'43' //LENGTH 6 STOMP EQU 5 DROP ARCHON L 9, =A(BEACON) NR 9,TREAS ST 9, =F'482'

```
                B           POINT+24
                LTORG
HOLE            DS          0D
POINT           DC          CL4’WIN’
                DC          (STOMP)A(BETA),V(FOOTBL)
                CLC         POINT-4, =F’482’          //LENGTH 6
                LA          1,1
                CVB         2,10(DEPUTY*TREAS)
                STC         3,POINT(TREAS*TREAS)
                BR          14
                    END
```

3 Make All the tables & Expand the Macro.

```
                    MACRO
        &LABEL  EXITLINK    &TYPE
                        AIF         ('&TYPE'  NE '').FOUND
                MNOTE       '**** MISSING TYPE PARAMETER'
                MEXIT
         .FOUND         ANOP
                AIF             ('&TYPE'  EQ 'N').NORMAL
                AIF             ('&TYPE'  EQ 'R').RETURN
                AIF         ('&TYPE'  EQ 'V').VALUE
                AIF         ('&TYPE'  EQ 'B').BOTH
                MEXIT
        .NORMAL     ANOP
                L           13,4(13)            PICK BACKWARD POINTER
                LM          14,12,12(13)        PICK ALL REGS
                BR          14                  RETURN TO CALLER
                AGO     .DONE
        .RETURN         ANOP
                L           13,4(13)            PICK BACKWARD POINTER
```

```
                L       14,12(13)       PICK ALL REGS EXCEPT
                LM      0,12,20(13)     REGISTER 15
                BR      14              RETURN TO CALLER
               AGO     .DONE
.VALUE         ANOP
                L       13,4(13)        PICK BACKWARD POINTER
                LM      14,15,12(13)    PICK ALL REGS EXCEPT
                LM      1,12,24(13)     REGISTER 0
                BR      14              RETURN TO CALLER
                AGO    .DONE
.BOTH          ANOP
                L       13,4(13)        PICK BACKWARD POINTER
                L       14,12(13)       PICK ALL REGS EXCEPT
                LM      1,12,24(13)     REGISTER 15 AND 0
                BR      14              RETURN TO CALLER
.DONE            ANOP
                MEND
MAIN            START
               EXITLINK Q
                EXITLINK
               EXITLINK V
                EXITLINK N
                EXITLINK R
               EXITLINK B
                END    MAIN
```

4	a) Explain One pass macro processor capable of handling macro calls within the macro definitions. Explain the stack operations with the help of suitable method.
5	Explain the Standard System Linkage criteria of an IBM 370 assembler.Write the minimum set of linkage instructions & then explain them one by one.
6	Use reductions and action routines as supplied . Generate the table that will come out after interpretation phase of the compiler.

BEGIN : Procedure (A,B);

DECLARE (A,B) FIXED BINARY (15) STATIC;

A = 10;

B = A;

END;

Action routines :

1. arg – place'.arg <idn> ' into matrix
2. assign – place' = <idn> <any>' into matrix
3. base_scale – place' base = binary' into proper identifier table entry
4. bgn_proc – place' . procbgn <idn> ' into matrix
5. end_proc – place' . procend' into matrix
6. error – print out error msg and top of stack
7. op_prec – parse arithmetic expression using rules of operator precedence leave result on stack
8. precision_class – place proper precision and STATIC into proper identifier table entry
9. return –put rtn into matrix
10. var_name – put name on temporary list

Reductions

1: / /***/

2: <idn> : PROCEDURE / bgn_proc /S1****/ 4

<any> <any> <any> / error/ S2S1 * / 2

4: (<idn>, <any> / arg / S4S1 **/ 4

5: (<idn>); / arg / " **/ 7

6: <any><any><any><any>/error/S2S1/7

7: DECLARE <any>// S!**/14

RETURN <any>/ return/ S1***/5

<idn> = / op_prec // 12

PROCEDURE END; / end_proc / " ***/2

<any> <any> / error/S1*/7

12: <idn> = <any>; / assign / "**/7

<any> <any> <any> <any>/ error / S2S1/7

14: <idn> FIXED BINARY / var_name base_scale / " *****/18

15: (<idn>, / var_name / S3 **/15

(<idn>)/ / S2 **/14

<any> <any>/ error / S2S1/7

18: (<lit>) STATIC, / precision_class/ "***/14

(<lit>) STATIC; / precision_class/ "**/7

<any> <any> <any> <any> <any>/ error / S2S1/7

7 a) Explain Relocating Loaders with their Transfer Vector Concept.

b) For the following Loader tables generate the program as much as possible.

RLD

ID	Flag	Length	Relative address
01	+	4	40
03	+	4	56
02	-	4	64
01	+	4	64

TXT

Opcode	Relative address	Address constant	Actual value
BALR	0		
SR	2		
L	4		
L	8		
ST	12		
BR	16		
DC	20		
DC	40	A(A+10)	66
DC	44		
DC	56	A(DELTA)	0
DC	60		
DC	64	A(STUDENT-	0

		SOLN)	

ESD

Symbol	Type	ID	Relative address	Length
Student	SD	01	0	68
A	LD			
SOLN	ER	02		
DELTA				

8

1. Is it always worthwhile to optimize a program?
2. Explain precisely the difference between the use of productions & reductions.
3. What Phases add to or modify the matrix, and what does each do? What phases reference the matrix without modifying it.
4. What information must be in the identifier table so that the code generation phase can generate the following statement

   X= A.B.C(50)
5. Give an example of a “permanent” table and an example of a “created” table.
6. In what phase is the elimination of common sub expressions performed? Why?
7. Give two examples of machine dependent optimization.
8. Is the use of automatic storage sufficient to eliminate the problem of recursion?