PART-B

1. Back Patching:

→ info is of labels

→ it indicates the address of the label in goto state. while producing TACs for gn exp.

→ 2 passes are used beco2 assigning the position of these labels are challenging

→ It can leave these addresses unidentified in 1st pass then populate in 2nd round.

→ the process of filling up gaps in incomplete transformation & info.

need for BP:

→ Boolean exp

→ flow of Control Statements

→ Labels & goto.

→ flow of prog in L & A.

→ Atleast one statement with L in scope of this g statement.

2. triples of exp a * -(b+c)

1: t1 := b+c → 1st triple, 2nd, 3rd.
2: t2 := - t1
3: t3 := a * t2

↓ temp variables to store intermediate results used to build up the final result of exp

3. x = *y : a = &x :

1: t1 := * y
2: x := t1
3: a := &x

4. idea behind generating 3-A-C during compilation.

→ to create I.C representation of S.C.
easily translated into Machine code
→ low-level rep. uses small no. of instructions

Benefits

Simplification: → break into small pieces.
→ simple to perform

Portability → platform independent representation
→ same code in diff platform.

Debugging → structured rep.

5. quadruples preferred over triples in an optimizing compiler?

→ When the statement are often moved around.
→ interposes an extra degree of indirection bet the computation of a value & its use

In triples notation,

Moving statement that defines a temporary value requires to change all pointers to that statement to in Arg1 & 2.
→ this makes difficult use in optimizing compiler.

6. notations to represent an intermediate lang.

TAC: → low-level representation of code.

Quadruples: → 4 field than TAC
→ Source 1, 2, Destination.

Syntax trees → hierarchical representation
→ relation bet exp & statements

Abstract Syntax tree:
→ Similar to Syntax tree
→ more abstract, omitting nodes that not affect the meaning of code

Bytecode:

→ virtual mach. (low-level rep of code)
Directly by processor.

7. how to solve the issues in the Design of code generator.

* I/p to code generator.

→ intermediate code generated from front end.

Linear rep. → postfix

TAC → Q, T, I

graphical rep. → syntax tree
→ DAG.

* target program

→ o/p of code generator

Absolute ML, Relocatable ML, Assembly lang.

* Memory Management.

Mapping name to address of Data Obj

* instruction selection.

→ Best instruct. improves efficiency.

* Register allocation issues
    → computation fastes.
    → Register Allocation & assignment.
* choice of evaluation order.
    → order in which the instructions are executed.

8. four principle use of registers in C. Generation

Registers → small, high-speed memory locations temporay store data & perform logic & Arith. exp/op.

Variable Storge:
    → Store values.

Parameta passing.
    → pass function arg bet func.

Return values.
    Retur. Val. from func.

temporay Storage.