Q1) Explain Input and Output functions in C.

Answer:

C language uses standard input/output functions defined in the header file **<stdio.h>**.

Two major functions used are:

1. printf() – Output function

- Used to **display text or values** on the screen.
- Syntax:
  printf("format string", variable_list);

Example:

printf("Hello");

printf("Value = %d", a);

Format specifiers include:

- %d → integer
- %f → float
- %c → character
- %s → string

2. scanf() – Input function

- Used to **take input from user via keyboard**.
- Syntax:
  scanf("format string", &variable_list);

Example:

int age;

scanf("%d", &age);

Note: & operator passes variable's address, required except for strings.

These functions enable interaction between user and program.

Q2) What is conditional execution? Explain if and if-else with examples.

Answer:

Conditional execution allows a program to **choose** whether to execute a block based on a condition.

if statement

Used when we want to execute statements only if condition is true.

Example:

```
if (marks >= 35) {
   printf("Pass");
}
```

if-else statement

Used when there are **two alternatives**.

Example:

```
if (marks >= 35)
   printf("Pass");
else
   printf("Fail");
```

If condition is true, first block executes; otherwise, second block executes.

Q3) What is an else-if ladder? Explain with an example.

Answer:

When multiple conditions must be tested, an a series of if-else is used.
This is called **else-if ladder**.

Example:

```
if (marks >= 90)
```

```
    printf("Grade A");
else if (marks >= 75)
    printf("Grade B");
else if (marks >= 60)
    printf("Grade C");
else if (marks >= 35)
    printf("Grade D");
else
    printf("Fail");
```

Only one block executes based on which condition is true first.

Q4) What is nested if? Give suitable example.

Answer:

When one if or else-if is written **inside another**, it forms a nested structure.

Example:

```
if (age >= 18) {
    if (citizen == 1)
        printf("Eligible to vote");
    else
        printf("Not a citizen");
} else {
    printf("Too young to vote");
}
```

Nested if allows comparison of multiple conditions together.

Q5) Write short note on switch statement.

Answer:

switch is a multi-way selection statement used to execute one block from many options based on the value of an expression.

Syntax:

```
switch(expression) {
   case value1: statements; break;
   case value2: statements; break;
   default: statements;
}
```

Example:

```
switch (choice) {
   case 1: printf("Start"); break;
   case 2: printf("Stop"); break;
   default: printf("Invalid");
}
```

It improves readability when checking a variable against multiple fixed values.

Q6) What is a loop? Explain while, do-while, and for loop in detail.

Answer:

Loop is a construct used to **repeat statements multiple times**.

1. while loop

- Entry-controlled loop (condition checked first).

- If condition false initially, loop may not run.

Example:

```
int i = 1;
while (i <= 5) {
   printf("%d ", i);
   i++;
}
```

2. do-while loop

- Exit-controlled loop (body executes before condition check).
- Executes **at least once**.

Example:

```
int i = 1;
do {
   printf("%d ", i);
   i++;
} while (i <= 5);
```

3. for loop

- Compact loop, ideal when number of iterations known.

Example:

```
for (int i = 1; i <= 5; i++) {
   printf("%d ", i);
}
```

Q7) What is a nested loop? Write a program to print 3×3 star pattern.

Answer:

A nested loop means **one loop inside another**.

Example program:

```
for (int i = 1; i <= 3; i++) {
    for (int j = 1; j <= 3; j++) {
        printf("* ");
    }
    printf("\n");
}
```

Output:

```
* * *
* * *
* * *
```

The outer loop controls rows; inner loop controls columns.

Q8) Explain break statement with example.

Answer:

break terminates the nearest loop or switch immediately.

Example:

```
for (int i = 1; i <= 10; i++) {
    if (i == 5)
        break;
    printf("%d ", i);
```

```
}
```

Output:

```
1 2 3 4
```

Q9) Explain continue statement with example.

Answer:

continue skips the current iteration and moves to next iteration.

Example:

```
for (int i = 1; i <= 5; i++) {
    if (i == 3)
        continue;
    printf("%d ", i);
}
```

Output:

```
1 2 4 5
```

Q10) Write a program to read a number and check whether it is Positive, Negative or Zero.

Answer:

```
#include<stdio.h>
void main()
{
    int n;
    printf("Enter number: ");
```

```
    scanf("%d", &n);

    if (n > 0)
        printf("Positive");
    else if (n < 0)
        printf("Negative");
    else
        printf("Zero");
}
```

Q11) Write a program to print numbers from 1 to 10 using for loop.

Answer:

```
#include<stdio.h>
void main()
{
    int i;
    for(i = 1; i <= 10; i++)
        printf("%d ", i);
}
```

Q12) Write a program to print Fibonacci series using loop.

Answer:

```
#include<stdio.h>

void main()

{

   int n, a = 0, b = 1, c, i;

   printf("Enter how many terms: ");

   scanf("%d", &n);


   for(i = 1; i <= n; i++) {

      printf("%d ", a);

      c = a + b;

      a = b;

      b = c;

   }

}
```

Q13) Differentiate between while and do-while loop.

Answer:

while loop	do-while loop
Condition checked first	Condition checked after body
May not execute even once	Executes at least once
Entry-controlled	Exit-controlled

Syntax: while(condition)	Syntax: do { } while(condition);

Q14) Explain goto statement. Why is it discouraged?

Answer:

goto is used to transfer control to a labelled part of program.

Example:

goto label;

...

label:

printf("Jumped");

However, goto creates unstructured and confusing code, making debugging difficult.
Therefore, its use is discouraged except in rare cases.

- **Explain all types of decision control statements in C with examples.**

What are Decision Control Statements?

In C, **decision control statements** are used when we want the program to **take decisions** based on some condition.

Example real life:

- If marks ≥ 35 → Pass
- Else → Fail
- if
- if-else
- else-if ladder
- nested if
- switch
- **conditional operator (?:)** (also a decision expression)

1. if Statement

Concept

if statement executes a block **only when the condition is true**.
If condition is false → body of if is skipped.

Syntax

```
if (condition) {
   // statements
}
```

Example

Program to check if a number is positive:

```
#include<stdio.h>
void main()
{
   int n;
   printf("Enter a number: ");
   scanf("%d", &n);
   if (n > 0) {
      printf("Number is Positive");
   }
   printf("\nEnd of program.");
}
```

If user enters 5 → “Number is Positive” print hoga.
Agar user enters -3 → if block skip ho jayega.

2. if-else Statement

Concept

Jab **do options** ho — one if condition is true, **else** if condition is false.

Example life:
“If it is raining, take umbrella, ELSE don’t take.”

Syntax

```
if (condition) {
   // executed if condition is true
} else {
   // executed if condition is false
}
```

Example

Check whether a student is **Pass or Fail**:

```
#include<stdio.h>
void main()
{
   int marks;
   printf("Enter marks: ");
   scanf("%d", &marks);
   if (marks >= 35) {
      printf("Pass");
   } else {
```

```
        printf("Fail");
    }
}
```

Condition true → “Pass”
False → “Fail”

3. else-if Ladder

Concept

Jab **multiple conditions** check karni ho (e.g., Grade system, menu, range, etc.), tab hum **else-if ladder** use karte hain.

Flow:

- Pehle if ka condition check hota hai.
- Agar false hua, else if 1 check hota hai.
- Fir next else if, and so on.
- Agar koi bhi true na ho → else execute hota hai.

Syntax

```
if (condition1) {
    // block 1
}
else if (condition2) {
    // block 2
}
else if (condition3) {
    // block 3
}
```

```
else {
   // default block
}
```

Example

Grading system based on marks:

```
#include<stdio.h>
void main()
{
   int marks;
   printf("Enter marks: ");
   scanf("%d", &marks);
   if (marks >= 90) {
      printf("Grade A");
   } else if (marks >= 75) {
      printf("Grade B");
   } else if (marks >= 60) {
      printf("Grade C");
   } else if (marks >= 35) {
      printf("Grade D");
   } else {
      printf("Fail");
   }
}
```

I

4. Nested if Statement

Concept

Jab ek if ke **andar** doosra if (ya if-else) ho, use **nested if** kehte hain.

Useful when:

- Multiple related conditions check karni ho
- E.g., age ke sath-sath citizenship, salary ke sath experience, etc.

Syntax

```
if (condition1) {
   // outer if
   if (condition2) {
      // inner if
   } else {
      // inner else
   }
} else {
   // outer else
}
```

Example

Check if a person is **eligible to vote** (age + citizen):

```
#include<stdio.h>
void main()
```

```
{
   int age;
   int citizen; // 1 = Indian, 0 = Not Indian
   printf("Enter age: ");
   scanf("%d", &age);
   printf("Is the person an Indian citizen? (1/0): ");
   scanf("%d", &citizen);
   if (age >= 18) {                  // outer if
      if (citizen == 1) {            // inner if
         printf("Eligible to vote");
      } else {
         printf("Not a citizen, cannot vote");
      }
   } else {
      printf("Not eligible due to age");
   }
}
```

5. switch Statement

Syntax

```
switch (expression) {
   case value1:
      statements;
      break;
```

```
    case value2:
        statements;
        break;
    ...
    default:
        statements;
}
```

Example – Menu driven program

```
#include<stdio.h>
void main()
{
    int choice;
    printf("1. Add\n2. Subtract\n3. Multiply\n");
    printf("Enter your choice: ");
    scanf("%d", &choice);
    switch (choice) {
        case 1:
            printf("You selected Addition");
            break;
        case 2:
            printf("You selected Subtraction");
            break;
```

```
        case 3:
            printf("You selected Multiplication");
            break;
        default:
            printf("Invalid choice");
    }
}
```

Important points

- switch expression **normally int or char type**.
- Each case ends **colon (:)**
- break compulsory.

6. Conditional Operator (?:)

Syntax

condition ? expression1 : expression2;

If condition true → expression1 execute.
Else → expression2 execute.

Example – Find maximum of two numbers:

```
#include<stdio.h>

void main()

{

   int a, b, max;

   printf("Enter two numbers: ");

   scanf("%d %d", &a, &b);

   max = (a > b) ? a : b;
```

```
    printf("Maximum = %d", max);
}
```

- (a > b) condition hai
- True → max = a
- False → max = b