Chapter 6 – Iterative Control Construct (Loops)

✅ Short Questions with Easy Answers

Q1. Define loop.

👉 A **loop** repeats one or more statements again and again until a condition is false.

Q2. Write two uses or advantages of loop.

1. Saves time – avoids writing the same code many times.
2. Helps to access a sequence of values (like printing 1 to 100).

Q3. Which program control statements are used to control iterations?

👉 while loop, do-while loop, for loop.

Q4. Which part of the loop contains the statements to be repeated?

👉 The **loop body**.

Q5. Which three steps must be done using loop control variable?

1. **Initialization** (start value)
2. **Test/Condition** (check when to stop)
3. **Increment/Decrement** (update value)

Q6. Define "while" loop.

👉 while executes statements **as long as condition is true**. Used when number of iterations is **not known in advance**.

Q7. Define "do-while" loop.

👉 do-while executes body **first,** then checks condition. It runs **at least once**.

Q8. Syntax of while loop with example.

```
int count = 0;
while (count < 5) {
   printf("C Programming\n");
   count++;
}
```

Q9. Syntax of do-while loop.

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

Q10. Syntax of for loop.

```
for(initialization; condition; increment/decrement) {
  // body
}
```

Example:

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

✅ **Long Questions (Easy + Examples)**

Q1. Explain working of while loop.

- Condition is **checked first**, then body executes.
- If condition is false at start, loop will **not run at all**.

👉 Example: Print numbers 1 to 5

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

Output: 1 2 3 4 5

Q2. Explain do-while loop.

- Body executes **first**, condition checked later.
- Always runs **at least once**.

👉 Example:

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

Output: 1 2 3 4 5

Q3. Explain working of for loop.

- Best when **number of iterations is known**.
- Has 3 parts: initialization, condition, update.

👉 Example: Print even numbers 2 to 10

```
for (int i = 2; i <= 10; i += 2) {
  printf("%d ", i);
}
```

Output: 2 4 6 8 10

Q4. Write a program that displays product of all odd numbers from 1 to 10 using for loop.

```
#include <stdio.h>
int main() {
  int product = 1;
  for (int i = 1; i <= 10; i += 2) {
    product *= i;   // multiply odd numbers
  }
  printf("Product = %d", product);
  return 0;
}
```

Odd numbers: 1, 3, 5, 7, 9
Output: Product = 945

Q5. Write a program that inputs an integer and display its table in descending order using for loop.

```
#include <stdio.h>
int main() {
  int n;
  printf("Enter a number: ");
  scanf("%d", &n);

  for (int i = 10; i >= 1; i--) {
    printf("%d x %d = %d\n", n, i, n*i);
  }
  return 0;
}
```

Input: 5
Output:

```
5 x 10 = 50
```

```
5 x 9 = 45
...
5 x 1 = 5
```

🔷 Extra External Questions with Answers

Q1. Difference between while and do-while loop?

- while → condition checked first, may run **0 times**.
- do-while → body runs first, so runs **at least once**.

Q2. Write a program to print sum of first 10 natural numbers using for loop.

```
int sum = 0;
for (int i = 1; i <= 10; i++) {
  sum += i;
}
printf("Sum = %d", sum);
```

Output: Sum = 55

Q3. Write a program to reverse numbers from 10 to 1 using while loop.

```
int i = 10;
while (i >= 1) {
  printf("%d ", i);
  i--;
}
```

Output: 10 9 8 7 6 5 4 3 2 1

Q4. What is a nested loop? Give example.
👉 A loop inside another loop.

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

Output: (1,1) (1,2) (1,3) (2,1) ... (3,3)