# Reading Comprehension

*C Programming Journey*

Reading comprehension questions related to for loops, while loops, and do-while loops in C:

1. What are loops used for in programming?
2. Differentiate between for loops, while loops, and do-while loops in C.
3. How does a for loop work in C? Provide an example.
4. Explain the syntax of a while loop in C. Give an example.
5. Describe the structure of a do-while loop in C. Provide a code example.
6. What is the primary difference between a while loop and a do-while loop?
7. How do you ensure a loop will execute at least once in C?
8. Discuss the potential risks associated with infinite loops and how to prevent them.
9. What is the purpose of the loop control variable in a for loop?
10. How do you terminate a loop prematurely in C?

# Reading Comprehension questions answers

1. **What are loops used for in programming?**

   - Loops are used in programming to execute a block of code repeatedly until a certain condition is met. They allow automation of repetitive tasks, saving time and effort.

2. **Differentiate between for loops, while loops, and do-while loops in C.**

   - A for loop is used when the number of iterations is known beforehand. A while loop is used when the condition is checked before the execution of the loop's body. A do-while loop is similar to a while loop, but it guarantees the execution of the loop's body at least once.

3. **How does a for loop work in C? Provide an example.**

    - A for loop in C consists of three parts: initialization, condition, and increment/decrement. It repeatedly executes a block of code as long as the condition remains true. Example:

```c
for (int i = 0; i < 5; i++) {
    printf("%d\n", i);
}
```

4. **Explain the syntax of a while loop in C. Give an example.**

    - The syntax of a while loop in C is:

```c
while (condition) {
    // code to be executed
}
```

Example:

```c
int i = 0;
while (i < 5) {
    printf("%d\n", i);
    i++;
}
```

5. **Describe the structure of a do-while loop in C. Provide a code example.**

    - A do-while loop in C executes the code block first and then checks the condition. It guarantees the execution of the block at least once. Example:

```c
int i = 0;
do {
    printf("%d\n", i);
    i++;
} while (i < 5);
```

6. **What is the primary difference between a while loop and a do-while loop?**
    - The primary difference is that a while loop checks the condition before executing the loop's body, while a do-while loop executes the body at least once and then checks the condition.
7. **How do you ensure a loop will execute at least once in C?**
    - To ensure a loop executes at least once, you use a do-while loop. It executes the block of code first and then checks the condition.
8. **Discuss the potential risks associated with infinite loops and how to prevent them.**
    - Infinite loops can lead to system freezes or crashes. To prevent them, ensure that the loop's condition eventually becomes false or use control statements like break.
9. **What is the purpose of the loop control variable in a for loop?**
    - The loop control variable in a for loop is used to control the number of iterations. It is initialized, checked against a condition, and incremented or decremented in each iteration.
10. **How do you terminate a loop prematurely in C?**
    - You can terminate a loop prematurely using the `break` statement. When encountered, `break` exits the loop immediately, regardless of the loop's condition.

# Lab exercises

1. **For Loops:**
    - Write a program to display the first 10 natural numbers using a for loop.
    - Create a program to calculate the factorial of a number using a for loop.
    - Develop a code to print the multiplication table of a given number using a for loop.
    - Implement a program to find the sum of even numbers between 1 to 100 using a for loop.

2. **While Loops:**

   - Write a program to find the sum of digits of a number using a while loop.
   - Create a program to reverse a number using a while loop.
   - Develop a code to check if a number is palindrome or not using a while loop.
   - Implement a program to calculate the Fibonacci series up to a given term using a while loop.

3. **Do-While Loops:**

   - Write a program to display the menu of a restaurant using a do-while loop and allow the user to choose options until they opt to exit.
   - Create a program to generate a random number between 1 to 100 and allow the user to guess it using a do-while loop.
   - Develop a code to calculate the average of numbers entered by the user until they enter a negative number using a do-while loop.
   - Implement a program to find the factorial of a number using a do-while loop.

# Lab exercises answers

1. **For Loops:**

   - Program to display the first 10 natural numbers:

```c
#include <stdio.h>
int main() {
    int i;
    printf("The first 10 natural numbers are:\n");
    for (i = 1; i <= 10; ++i) {
        printf("%d ", i);
    }
    return 0;
}
```

   - Program to calculate the factorial of a number:

```c
#include <stdio.h>
int main() {
    int num, i;
    unsigned long long factorial = 1;
    printf("Enter a positive integer: ");
    scanf("%d", &num);
    for (i = 1; i <= num; ++i) {
        factorial *= i;
    }
    printf("Factorial of %d = %llu", num, factorial);
    return 0;
}
```

- Program to print the multiplication table of a given number:

```c
#include <stdio.h>
int main() {
    int num, i;
    printf("Enter an integer: ");
    scanf("%d", &num);
    for (i = 1; i <= 10; ++i) {
        printf("%d * %d = %d\n", num, i, num * i);
    }
    return 0;
}
```

- Program to find the sum of even numbers between 1 to 100:

```c
#include <stdio.h>
int main() {
    int sum = 0, i;
    for (i = 2; i <= 100; i += 2) {
        sum += i;
    }
    printf("Sum of even numbers between 1 to 100: %d", sum);
    return 0;
}
```

2. **While Loops:**

- Program to find the sum of digits of a number:

```c
#include <stdio.h>
int main() {
    int num, digit, sum = 0;
    printf("Enter a number: ");
    scanf("%d", &num);
    while (num != 0) {
        digit = num % 10;
        sum += digit;
        num /= 10;
    }
    printf("Sum of digits: %d", sum);
    return 0;
}
```

- Program to reverse a number:

```
#include <stdio.h>
int main() {
    int num, reversedNum = 0, remainder;
    printf("Enter an integer: ");
    scanf("%d", &num);
    while (num != 0) {
        remainder = num % 10;
        reversedNum = reversedNum * 10 + remainder;
        num /= 10;
    }
    printf("Reversed number: %d", reversedNum);
    return 0;
}
```

- Program to check if a number is palindrome:

```
#include <stdio.h>
int main() {
    int num, reversedNum = 0, originalNum, remainder;
    printf("Enter an integer: ");
    scanf("%d", &num);
    originalNum = num;
    while (num != 0) {
        remainder = num % 10;
        reversedNum = reversedNum * 10 + remainder;
        num /= 10;
    }
    if (originalNum == reversedNum) {
        printf("%d is a palindrome.", originalNum);
    } else {
        printf("%d is not a palindrome.", originalNum);
    }
    return 0;
}
```

- Program to calculate the Fibonacci series up to a given term:

```
#include <stdio.h>
int main() {
    int n, first = 0, second = 1, next, c;
    printf("Enter the number of terms: ");
    scanf("%d", &n);
    printf("Fibonacci series: ");
    for (c = 0; c < n; c++) {
        printf("%d, ", first);
        next = first + second;
        first = second;
        second = next;
    }
    return 0;
}
```

3. **Do-While Loops:**

   - Program to display the menu of a restaurant:

```
#include <stdio.h>
int main() {
    int choice;
    do {
        printf("Menu:\n1. Pizza\n2. Burger\n3. Pasta\n4. Exit\n");
        printf("Enter your choice: ");
        scanf("%d", &choice);
        switch (choice) {
            case 1:
                printf("You ordered Pizza.\n");
                break;
            case 2:
                printf("You ordered Burger.\n");
                break;
            case 3:
                printf("You ordered Pasta.\n");
                break;
            case 4:
                printf("Exiting...\n");
                break;
            default:
                printf("Invalid choice!\n");
        }
    } while (choice != 4);
    return 0;
}
```

   - Program to generate a random number and allow the user to guess it:

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
int main() {
    int guess, randomNumber;
    srand(time(0));
    randomNumber = rand() % 100 + 1;
    do {
        printf("Guess the number (1-100): ");
        scanf("%d", &guess);
        if (guess > randomNumber) {
            printf("Too high! Try again.\n");
        } else if (guess < randomNumber) {
            printf("Too low! Try again.\n");
        } else {
            printf("Congratulations! You guessed it right.\n");
        }
    } while (guess != randomNumber);
    return 0;
}
```

- Program to calculate the average of numbers entered by the user until they enter a negative number:

```
#include <stdio.h>
int main() {
    double num, sum = 0;
    int count = 0;
    do {
        printf("Enter a number (Negative to stop): ");
        scanf("%lf", &num);
        if (num >= 0) {
            sum += num;
            count++;
        }
    } while (num >= 0);
    printf("Average: %.2lf", sum / count);
    return 0;
}
```

- Program to find the factorial of a number:

```
#include <stdio.h>
int main() {
    int num, factorial = 1;
    printf("Enter a positive integer: ");
    scanf("%d", &num);
    do {
        factorial *= num;
        num--;
    } while (num > 0);
    printf("Factorial: %d", factorial);
    return 0;
}
```

*C Programming Journey*