Name: aman qasim

ROLL NO: bscs-gill-fa22-004

Course: Programming Fundamentals

Semester: BSCS, BSSE (1st semester)

Q1. Define the following terms in the context of programming: (5 marks)

a) Algorithm:

An algorithm is a step-by-step procedure or set of rules for solving a problem or performing a task. It is the logical sequence of instructions that must be followed to achieve a desired output from a given input.

b) Variable:

A variable is a named storage location in a program that holds a value, which can be modified during the program's execution. It is used to store data such as numbers, characters, or other information.

c) Data Type:

A data type defines the type of data a variable can store. It specifies the kind of value that can be assigned to the variable, such as integers, floating-point numbers, characters, etc. Examples include int, float, char, and string.

d) Function:

A function is a block of code that performs a specific task. It

can take inputs (parameters) and return an output (result). Functions allow code reuse and help in organizing the program into manageable pieces.

e) Loop:

A loop is a control structure that repeats a block of code as long as a certain condition is met. The common types of loops include for, while, and do-while. Loops are used to execute a set of statements multiple times without repeating code.

Q2. What is the difference between procedural programming and object-oriented programming? Provide examples of each. (5 marks)

- Procedural Programming:

o Focuses on a sequence of procedures (functions or routines) that operate on data.

o In procedural programming, the program is divided into smaller procedures or functions that take inputs, process them, and produce outputs.

o Example: A program where a series of functions are called to perform tasks like reading user input, calculating results, and displaying output.

Example of procedural programming (C++):

#include

```
void greet() {
std::cout << "Hello, World!" << std::endl;
}

int main() {
greet();
return 0;
}
```

- Object-Oriented Programming (OOP):
  - Focuses on objects, which are instances of classes. A class is a blueprint that defines properties (attributes) and methods (functions) for the objects.
  - Data and functions are encapsulated in objects. OOP emphasizes concepts like inheritance, polymorphism, abstraction, and encapsulation.
  - Example: A program that defines a class to represent an entity (like a car), and objects of that class perform actions related to the car.

Example of OOP (C++):

```
#include

class Car {
public:
```

```
std::string brand;
int speed;

void accelerate() {
speed += 10;
std::cout << "Speed is now: " << speed << " km/h"
<< std::endl;
    }
};

int main() {
Car myCar;
myCar.brand = "Toyota";
myCar.speed = 0;
myCar.accelerate();
return 0;
}
```

Q3. Explain the concept of control structures in C++ programming. Discuss the three main types of control structures with examples. (5 marks)

Control structures in C++ are used to control the flow of execution based on certain conditions. The three main types are:

1. Sequential Control Structure:

The default mode of execution where statements are executed in sequence, one after another.

Example:

```
int main() {
int a = 10, b = 20;
int sum = a + b;      //sequential execution
std::cout << "Sum: " << sum << std::endl;
return 0;
}
```

2. Selection (Conditional) Control Structure:

Used for making decisions. The program can take different paths based on conditions (e.g., if, else, switch).

Example (if-else):

```
int main() {
int num = 5;
if (num > 0) {
std::cout << "Positive" << std::endl;
   else {
std::cout << "Negative or zero" << std::endl;
        }
return 0;
}
```

3. Iteration (Looping) Control Structure:

Used to repeat a block of code multiple times as long as a certain condition holds true. Common loops include for, while, and do-while.

Example (for loop):

```
int main() {
for (int i = 1; i <= 5; i++) {
std::cout << "Iteration " << i << std::endl;
        }
return 0;
}
```

Q4. What is debugging in programming? Explain some common debugging techniques that can be used to find and fix errors in a program. (5 marks)

Debugging is the process of identifying, analyzing, and fixing bugs (errors) in a program. Common techniques include:

1. Print Statements:

Inserting std::cout statements to display variable values or check the program's flow at certain points.

2. Using a Debugger:

A debugger allows you to step through the code line by line,

inspect variable values, and set breakpoints to pause execution at specific points.

3. Code Review:

Reviewing the code manually or with a peer to identify logical or syntax errors.

4. Unit Testing:

Writing small test cases to check if individual parts of the program (functions) work as expected.

5. Rubber Duck Debugging:

Explaining the code or problem to someone (or something) else, which often helps clarify the issue.

Q5. Describe the difference between call by value and call by reference in function passing. Provide a C++ code example to illustrate both methods. (5 marks)

- Call by Value:

In this method, a copy of the argument is passed to the function. Any changes made to the parameter inside the function do not affect the original variable.

Example:

```
void byValue(int x) {
x = 10;    //anges only the local copy
}
```

```
int main() {
int a = 5;
byValue(a);
std::cout << "value of a: " << a << std::endl;     //
output: 5
return 0;
}
```

- Call by Reference:

In this method, a reference (memory address) of the argument is passed to the function. Any changes made to the parameter affect the original variable.

Example:

```
void byReference(int &x) {
x = 10;     //anges the original variable
}

int main() {
int a = 5;
byReference(a);
std::cout << "value of a: " << a << std::endl;     //
output: 10
return 0;
}
```

Q6. Write a C++ program that accepts a number from the user and checks whether it is positive, negative, or zero. (5 marks)

```
#include

int main() {
int num;
std::cout << "Enter a number: ";
std::cin >> num;

if (num > 0) {
std::cout << "The number is positive." << std::endl;
   else if (num < 0) {
std::cout << "The number is negative." << std::endl;
   else {
std::cout << "The number is zero." << std::endl;
        }

return 0;
}
```

Q7. Write a C++ program to find the sum of all even numbers between 1 and a given number (inclusive). (5 marks)

```
#include

int main() {
int n, sum = 0;
std::cout << "Enter a number: ";
std::cin >> n;

for (int i = 2; i <= n; i += 2) {
sum += i;
        }

std::cout << "The sum of even numbers between 1 and "
<< n << " is: " << sum << std::endl;
return 0;
}
```

Q8. Write a C++ program to display the Fibonacci series up to a certain number entered by the user. (5 marks)

```
#include
```

```
int main() {
int n, a = 0, b = 1, next;
std::cout << "Enter the number of terms: ";
std::cin >> n;

std::cout << "Fibonacci Series: ";
for (int i = 1; i <= n; i++) {
std::cout << a << " ";
next = a + b;
a = b;
b = next;
        }
std::cout << std::endl;
return 0;
}
```

Q9. Write a C++ program that reads a character from the user and checks if the character is a vowel or consonant. (5 marks)

```
#include
#include

int main() {
```

```
char ch;
std::cout << "Enter a character: ";
std::cin >> ch;

ch = tolower(ch);    // Convert to lowercase to simplify comparison

if (ch == 'a' || ch == 'e' || ch == 'i' || ch == 'o' || ch == 'u') {
std::cout << ch << " is a vowel." << std::endl;
    else {
std::cout << ch << " is a consonant." << std::endl;
        }

return 0;
}
```

Q10. Write a C++ program to calculate the factorial of a number using both iterative and recursive methods. (5 marks)

- Iterative method:

```
#include

int factorialIterative(int n) {
```

```
int result = 1;
for (int i = 1; i <= n; i++) {
result *= i;
        }
return result;
}

int main() {
int num;
std::cout << "Enter a number: ";
std::cin >> num;

std::cout << "Factorial (Iterative): " <<
factorialIterative(num) << std::endl;
return 0;
}
```

- Recursive method:

```
#include

int factorialRecursive(int n) {
if (n == 0 || n == 1)
return 1;
return n * factorialRecursive(n - 1);
}
```