VICTORIA UNIVERSITY OF BANGLADESH

 **FINAL ASSESSMENT**

SUB: Computer Fundamentals and Programming Techniques.

SUB. CODE: CSE-108

SEMESTER: Spring 2024

STUDENT NAME: Easha Moni

STUDENT ID: 1522550021

BATCH: 55th

PROGRAM: BTHM

**Question no 02(a)**

Write a C- program to find the year is 'Leap Year or not'.

**Answer to the question no 02(a)**

Sure, here's a simple C program to determine whether a given year is a leap year or not:

This program prompts the user to input a year and then checks if it's divisible by 4. If it is, it further checks if it's not divisible by 100 or if it's divisible by 400. If either condition is true, it prints that the year is a leap year; otherwise, it prints that it's not a leap year.

**Question no 02(b)**

write an algorithm and draw a flow chart to find the biggest from the given three scores.

**Answer to the question no 02(b)**

Here's an algorithm to find the biggest from three given scores:

Algorithm to Find the Biggest of Three Scores:

1. Start

2. Input three scores: score1, score2, score3

3. Set maxScore toscore1

4. If score2 > maxScore, set maxScore to score2

5. If score3 > maxScore, set maxScore to score3

6. Output maxScore as the biggest score

7. End

And here's a simple flowchart representing the algorithm:



This flowchart represents the steps mentioned in the algorithm. It starts by inputting the three scores, then it compares each score with the current maximum score and updates it if necessary. Finally, it outputs the maximum score and ends the program.

**Question no 03(a)**

Draw the basic gates and write down the law for getting output from the given input where maximum input is two only.

**Answer to the question no 03(a)**

Sure, here are the basic logic gates along with their symbols and truth tables for two inputs:

1. **AND Gate**:

Symbol:

\_\_\_\_ | | | AND | |\_\_\_\_\_\_|

Truth Table:

|  |  |  |
| --- | --- | --- |
| Input A | Input B | Output |
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |

Output Law: Output is 1 only when both inputs are 1.

1. **OR Gate**:

Symbol:

\_\_\_\_ | | | OR | |\_\_\_\_\_\_

Truth Table:

|  |  |  |
| --- | --- | --- |
| Input A | Input B | Output  |
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 1 |

Output Law: Output is 1 when at least one input is 1.

1. **NOT Gate**:

Symbol:

\_\_\_ | | | NOT| |\_\_\_|

Truth Table:

|  |  |
| --- | --- |
| Input  | Output  |
| 0 | 1 |
| 1 | 0 |

Output Law: Output is the inverse of the input.

These are the basic logic gates and their corresponding truth tables for two inputs. The output laws described for each gate summarize the conditions under which the output will be 1.

**Question no 03(b)**

write a c-program to find the average from the given five numbers.

**Answer to the question no 03(b)**

Certainly! Here's a simple C program to find the average of five given numbers:

‘int main () { int num1, num2, num3, num4, num5; float average;

// Input five numbers

 Printf ("Enter five numbers: ");

scanf("%d %d %d %d %d", &num1, &num2, &num3, &num4, &num5);

// Calculate the average average = (float)(num1 + num2 + num3 + num4 + num5) / 5;

// Display the average

 printf("The average of %d, %d, %d, %d, and %d is: %.2f\n", num1, num2, num3, num4, num5, average); return 0; }’

This program prompts the user to input five numbers, calculates their sum, and then divides the sum by 5 to find the average. Finally, it displays the average with two decimal places.

**Question no 04(a)**

Write an algorithm and flow chart to find the summation from then given series such as 1+2+3+…….+N

**Answer to the question no 04(a)**

Here's an algorithm to find the summation of the series 1+2+3+…+N:

Algorithm to Find Summation of Series:

1. Start

2. Input N (the last number in the series)

3. Set sum to 0

4. Initialize a variable i to 1

5. Repeat steps 6 to 8 until i is less than or equal to N

6. Add i to sum

7. Increment i by 1

8. End of loop

9. Output sum as the summation of the series

10. End

And here’s a flowchart representing the algorithm:



This flowchart represents the steps mentioned in the algorithm. It starts by inputting the value of N, sets the sum to 0, and initializes a variable i to 1. Then, it iterates through each number from 1 to N, adding it to the sum. Finally, it outputs the sum and ends the program.

**Question no 04(b)**

Write a C-program to for the QN-4(a).

**Answer to the question no 04(b)**

Certainly! Assuming QN-4(a) refers to the task of finding the summation of the series 1+2+3+…+N1+2+3+…+N, here's the C program:

Int main() { int N, sum = 0; // Input the value of N

printf("Enter the value of N: ");

scanf("%d", &N);

// Calculate the summation of the series for (int i = 1; i <= N; i++) { sum += i; }

// Display the summation

printf("The summation of the series 1 + 2 + ... + %d is: %d\n", N, sum); return 0; }

This program prompts the user to input the value of N, then it uses a for loop to iterate from 1 to N and adds each number to the variable sum. Finally, it displays the summation of the series.

**Question no 05(a)**

Define Algorithm and flow-chart.

**Answer to the question no 05(a)**

Sure, here are definitions for both terms:

1. **Algorithm**: An algorithm is a step-by-step procedure or set of rules used for solving a computational problem. It's a finite sequence of well-defined instructions, typically used for performing a computation, carrying out data processing, or performing other tasks. Algorithms can be expressed in various ways, including natural language, pseudocode, flowcharts, or programming languages. They serve as the foundation for developing software and solving complex problems efficiently.
2. **Flowchart**: A flowchart is a graphical representation of an algorithm or process. It uses different shapes and arrows to depict the steps and decision points in the algorithm. Each shape represents a specific type of instruction or action, such as input/output, processing, or decision making. Flowcharts are widely used in various fields, including computer programming, engineering, business, and education, as they provide a visual way to understand and communicate complex processes. They are particularly useful for visualizing the flow of control in algorithms and identifying potential areas for optimization or improvement.

In summary, an algorithm provides a logical sequence of steps to solve a problem, while a flowchart visually represents these steps using symbols and arrows to illustrate the flow of control.

**Question no 05(b)**

Write down the basic difference between ALGORITHM and FLOW-CHART.

**Answer to the question no 05(b)**

Certainly! Let's delve deeper into the distinctions between algorithms and flowcharts:

**Algorithm:** An algorithm is a precise, step-by-step set of instructions or rules designed to solve a problem or accomplish a task. It's essentially a blueprint or a recipe for solving a particular computational problem. Algorithms are often expressed in natural language, pseudocode, or a specific programming language. Here are some key points about algorithms:

1. **Abstract Concept**: An algorithm is an abstract concept. It defines the logic and sequence of operations required to solve a problem but doesn't specify how those operations are to be carried out in a particular programming language or notation.
2. **Language Independent**: Algorithms are not tied to any specific programming language. They can be implemented in various programming languages or even represented in natural language without affecting their fundamental logic.
3. **Focus on Logic**: The primary focus of an algorithm is on the logical flow of operations rather than the specific implementation details. It describes the "what" of the solution rather than the "how."
4. **Problem Solving**: Algorithms are used to solve a wide range of computational problems, from simple arithmetic calculations to complex optimization tasks and everything in between.

**Flowchart:** A flowchart is a graphical representation of an algorithm or a process. It uses standardized symbols and connectors to visually depict the sequence of steps and decision points involved in solving a problem or completing a task. Here are some key points about flowcharts:

1. **Visual Representation**: Flowcharts provide a visual representation of an algorithm or a process. They use symbols such as rectangles, diamonds, circles, and arrows to represent different types of operations, decisions, and connections between steps.
2. **Sequential Structure**: Flowcharts typically have a sequential structure, with steps arranged in a top-to-bottom or left-to-right manner to represent the order in which they are executed.
3. **Decision Points**: Flowcharts include decision points where the flow of execution can branch into different paths based on certain conditions or criteria. These decision points are represented using diamond-shaped symbols.
4. **Communication Tool**: Flowcharts serve as effective communication tools, allowing programmers, analysts, and stakeholders to visualize and understand the logical flow of an algorithm or a process. They can be used for planning, documentation, and troubleshooting.

In summary, while algorithms define the logical steps for solving a problem, flowcharts provide a visual representation of those steps, making them easier to understand and communicate. Algorithms are abstract concepts, whereas flowcharts are concrete visualizations of those concepts. Both are essential tools in computer science and problem-solving, each serving its own purpose in the software development lifecycle.

**Question no 05(c)**

Define ROM, RAM, Cache and Flash memory.

**Answer to the question no 05(c)**

Here are definitions for each:

**ROM (Read-Only Memory):** ROM is a type of non-volatile memory that retains its data even when the power is turned off. It contains firmware or permanent software instructions that are essential for booting up the computer or electronic device. Unlike RAM, ROM is read-only, meaning that its contents cannot be easily modified or overwritten. It is used to store critical system software, such as the BIOS (Basic Input/Output System) in computers, which initializes hardware components during the boot-up process.

**RAM (Random Access Memory):** RAM is a type of volatile memory that is used by computers and other electronic devices to store data and program instructions temporarily. Unlike ROM, RAM loses its stored information when the power is turned off or the device is restarted. RAM is much faster to read from and write to compared to other storage mediums like hard disk drives (HDDs) or solid-state drives (SSDs). It serves as the main memory for the CPU (Central Processing Unit), providing quick access to data and instructions needed for ongoing tasks and processes.

**Cache Memory:** Cache memory is a small, high-speed type of volatile or sometimes non-volatile memory that is located between the CPU and the main memory (RAM) in a computer system. Its primary purpose is to store frequently accessed data and instructions to reduce the time it takes for the CPU to access them. By keeping a copy of frequently used data closer to the CPU, cache memory helps improve overall system performance by reducing latency and speeding up data retrieval. There are typically several levels of cache memory in modern computer architectures, with each level offering progressively faster access speeds but smaller storage capacities.

**Flash Memory:** Flash memory is a type of non-volatile memory that can be electrically erased and reprogrammed. It is commonly used in electronic devices such as USB flash drives, SSDs, memory cards, and smartphones to store data persistently. Unlike traditional magnetic storage media like hard disk drives (HDDs), flash memory has no moving parts, making it more durable and resistant to mechanical failure. It is based on a technology called floating-gate transistors, which trap electric charge to represent binary data (0s and 1s). Flash memory is known for its fast read and write speeds, low power consumption, and compact form factor, making it well-suited for portable and embedded applications.