**DAILY ASSESSMENT FORMAT**

|  |  |  |  |
| --- | --- | --- | --- |
| **Date:** | **22/06/2020** | **Name:** | **Yashaswini.R** |
| **Course:** | **C++ programming** | **USN:** | **4AL17EC098** |
| **Topic:** | 1. **Module 1 : Basic concepts** 2. **Module 2 : Conditions and loops** | **Semester & Section:** | **6th Sem ‘B’ Sec** |
| **GitHub Repository:** | **Yashaswini** |  |  |

|  |  |
| --- | --- |
| **FORENOON SESSION DETAILS** | |
| **Image of session** | |
| **Comments**  **Comments** are explanatory statements that you can include in the C++ code to explain what the code is doing.  The compiler ignores everything that appears in the comment, so none of that information shows in the result.   A comment beginning with **two slashes (//)**is called a single-line comment. The slashes tell the compiler to ignore everything that follows, until the end of the line.  **Multi-Line Comments**  Comments that require multiple lines begin with **/\*** and end with **\*/** You can place them on the same line or insert one or more lines between them.  /\* This is a comment \*/ /\* C++ comments can span multiple lines \*/  **Using Comments**  Comments can be written anywhere, and can be repeated any number of times throughout the code. Within a comment marked with /\* and \*/, // characters have no special meaning, and vice versa. This allows you to "nest" one comment type within the other.  /\* Comment out printing of Hello world!  cout << "Hello world!"; // prints Hello world!  \*/  **Variables**  Creating a **variable**reserves a memory location, or a space in memory for storing values. The compiler requires that you provide a **data type** for each variable you declare.  C++ offer a rich assortment of built-in as well as user defined **data types**.  **Integer**, a built-in type, represents a whole number value. Define integer using the keyword **int**. C++ requires that you specify the **type**and the **identifier**for each variable defined.An **identifier**is a name for a variable, function, class, module, or any other user-defined item. An identifier starts with a letter (A-Z or a-z) or an underscore (\_), followed by additional letters, underscores, and digits (0 to 9).  **Operator Precedence**  Operator **precedence**determines the grouping of terms in an expression, which affects how an expression is evaluated. Certain operators take higher precedence over others; for example, the multiplication operator has higher precedence over the addition operator. **For example:**  int x = 5+2\*2; cout << x; // Outputs **9**  The program above evaluates 2\*2 first, and then adds the result to 5.  As in mathematics, using **parentheses**alters operator precedence.  int x = (5 + 2) \*2; cout << x;  // Outputs 14  **Assignment Operators**  The simple **assignment**operator (=) assigns the right side to the left side.   C++ provides shorthand operators that have the capability of performing an operation and an assignment at the same time.  **For example:**int x = 10; **x += 4**; // equivalent to x = x + 4 **x -= 5**; // equivalent to x = x - 5  The same shorthand syntax applies to the multiplication, division, and modulus operators.x \*= 3; // equivalent to x = x \* 3 x /= 2; // equivalent to x = x / 2 x %= 4; // equivalent to x = x % 4  **Increment Operator**  The **increment**operator is used to increase an integer's value by one, and is a commonly used C++ operator. **x++;** //equivalent to x = x + 1  **Decision Making**  The **if** statement is used to execute some code if a condition is true.  **Syntax:if** (condition) { statements } The **condition**specifies which expression is to be evaluated. If the condition is true, the statements in the curly brackets are executed.  **Example:**  if (**10 == 10**) { cout << "Yes"; }  // Outputs "Yes" else Statement  example: int mark = 90;  if (mark < 50) { cout << "You failed." << endl; cout << "Sorry" << endl; } else { cout << "Congratulations!" << endl; cout << "You passed." << endl; cout << "You are awesome!" << endl; }  /\* Outputs Congratulations! You passed. You are awesome! \*/ Nested if else Statements C++ provides the option of nesting an unlimited number of if/else statements. **For example:**  int age = 18; if (age > 14) { if(age >= 18) { cout << "Adult"; } else { cout << "Teenager"; } } else { if (age > 0) { cout << "Child"; } else { cout << "Something's wrong"; } } | |
|  |