**DAILY ASSESSMENT FORMAT**

|  |  |  |  |
| --- | --- | --- | --- |
| **Date:** | **24-06-2020** | **Name:** | **Kiran N** |
| **Course:** | **C++** | **USN:** | **4al16ec031** |
| **Topic:** | **•Classes and objects**  **•More on classes** | **Semester & Section:** | **8th and A** |
| **Github Repository:** | **Kiran-course** |  |  |

|  |
| --- |
| **FORENOON SESSION DETAILS** |
| C++ Classes and Objects  Class:  A class in C++ is the building block, that leads to Object-Oriented programming. It is a user-defined data type, which holds its own data members and member functions, which can be accessed and  used by creating an instance of that class. A C++ class is like a blueprint for an object.  For Example: Consider the Class of Cars. There may be many cars with different names and brand but  all of them will share some common properties like all of them will have 4 wheels,Speed Limit,  Mileage range etc. So here, Car is the class and wheels, speed limits, mileage are their properties.  • A Class is a user defined data-type which has data members and member functions.  •Data members are the data variables and member functions are the functions used to manipulate  these variables and together these data members and member functions defines the properties and  behavior of the objects in a Class.  •In the above example of class Car , the data member will be speed limit, mileage etc and member  functions can be apply brakes, increase speed etc.  An Object is an instance of a Class. When a class is defined, no memory is allocated but when it is  instantiated (i.e. an object is created) memory is allocated.  Everything in C++ is associated with classes and objects, along with its attributes and methods. For  example: in real life, a car is an object. The car has attributes, such as weight and color, and methods,  such as drive and brake.  Attributes and methods are basically variables and functions that belongs to the class. These are often  referred to as "class members".  A class is a user-defined data type that we can use in our program, and it works as an object constructor, or a "blueprint" for creating objects.  Create a Class To create a class, use the class keyword:  Example  Create a class called "MyClass":  Class MyClass { // The class  Public : // Access specifier  Int myNum; // Attribute (int variable)  string myString;  // Attribute (string variable)  };  Create an Object  In C++, an object is created from a class. We have already created the class named MyClass , so now we can use this to create objects.  To create an object of MyClass, specify the class name, followed by the object name.  To access the class attributes (myNum and myString), use the dot syntax (.) on the object  Example  Create an object called "myObj" and access the attributes:  Class MyClass { // The class  Public : // Access specifier  Int myNum; // Attribute (int variable)  string myString; // Attribute (string variable)  };  Int main() { MyClass myObj;  // Create an object of MyClass  // Access attributes and set values  myObj.myNum=15;  myObj.myString="Some text"; // Print attribute values  cout << myObj.  myNum<<"\n";  cout << myObj.myString;  return 0;  }  Abstraction in C++  Data abstraction is one of the most essential and important feature of object oriented programming in C++. Abstraction means displaying only essential information and hiding the details. Data abstraction refers to providing only essential information about the data to the outside world, hiding the background details or implementation.  Consider a real life example of a man driving a car. The man only knows that pressing the accelerators  will increase the speed of car or applying brakes will stop the car but he does not know about how on  pressing accelerator the speed is actually increasing, he does not know about the inner mechanism of the car or the implementation of accelerator, brakes etc in the car. This is what abstraction is.  Abstraction using Classes:  We can implement Abstraction in C++ using classes. Class helps us to group data members and member functions using available access specifiers. A Class can decide which data member will be visible to outside world and which is not.  Encapsulation in C++ In normal terms Encapsulation  is defined as wrapping up of data and information under a single unit. In Object Oriented Programming, Encapsulation is defined as binding together the data and the functions that  manipulates them.  Consider a real life example of encapsulation, in a company there are different sections like the accounts section, finance section, sales section etc. The finance section handles all the financial transactions and keep records of all the data related to finance. Similarly the sales section handles all the sales related activities and keep records of all the sales. Now there may arise a situation when for some reason an official from finance section needs all the data about sales in a particular month. In this case, he is not allowed to directly access the data of sales section. He will first have to contact some other officer in the sales section and then request him to give the particular data. This is what encapsulation is. Here the data of sales section and the employees that can manipulate them are wrapped under a single name “sales section”.  Constructors in C++  What is constructor?  A constructor is a member function of a class which initializes objects of a class. In C++, Constructor is  automatically called when object(instance of class) create. It is special member function of the class.  How constructors are different from a normal member function?  A constructor is different from normal functions in following ways  •Constructor has same name as the class itself  •Constructors don’t have return type  •A constructor is automatically called when an object is created.  •If we do not specify a constructor, C++ compiler generates a default constructor for us (expects  no parameters and has an empty body). |