**DAILY ASSESSMENT FORMAT**

|  |  |  |  |
| --- | --- | --- | --- |
| **Date:** | **25-June-2020** | **Name:** | **Raziya Banu** |
| **Course:** | **C++ Programming** | **USN:** | **4AL16EC058** |
| **Topic:** | **Inheritance and polymorphism** | **Semester & Section:** | **8th sem & ‘B’ section** |
| **Github Repository:** |  |  |  |

|  |
| --- |
| **FORENOON SESSION DETAILS** |
| **Image of session** |
| **Report –**  In my first session today I have studied about - **Inheritance and polymorphism**  [**Inheritance**](https://www.geeksforgeeks.org/inheritance-in-c/)**:**  Inheritance is one in which a new class is created that inherits the properties of the already exist class. It supports the concept of code reusability and reduces the length of the code in object-oriented programming.  Types of Inheritance are:   1. Single inheritance 2. Multi-level inheritance 3. Multiple inheritance 4. Hybrid inheritance 5. Hierarchical inheritance   **Example of Inheritance:**   * C++ * Java   filter\_none  edit  play\_arrow  brightness\_4   |  | | --- | | #include "iostream"  using namespace std;    class A  {      int a, b;      public:      void add(int x, int y)      {          a=x;b=y;          cout<<(a+b)<<endl;      }  };    class B : public A  {      public:      void print(int x, int y)      {          add(x, y);      }  };    int main() {     B b1;     b1.print(5, 6);  } |   **Output:**  addition of a+b is:11  Here, class B is the derived class which inherit the property(**add method**) of the base class A. |

|  |  |  |  |
| --- | --- | --- | --- |
| **Date:** | **25-June-2020** | **Name:** | **Raziya Banu** |
| **Course:** | **C++ Programming** | **USN:** | **4AL16EC058** |
| **Topic:** | **Polymorphism** | **Semester & Section:** | **8th sem & ‘B’ section** |
| **AFTERNOON SESSION DETAILS** | | | |
| **Image of session** | | | |
| [**Polymorphism**](https://www.geeksforgeeks.org/polymorphism-in-c/)**:** Polymorphism is that in which we can perform a task in multiple forms or ways. It is applied to the functions or methods. Polymorphism allows the object to decide which form of the function to implement at compile-time as well as run-time.  Types of Polymorphism are:   1. Compile-time polymorphism (Method overloading) 2. Run-time polymorphism (Method Overriding)   **Example of Polymorphism:**   * C++ * Java   filter\_none  edit  play\_arrow  brightness\_4   |  | | --- | | #include "iostream"  using namespace std;    class A  {      int a, b, c;      public:      void add(int x, int y)      {          a=x;b=y;          cout<<"addition of a+b is:"<<(a+b)<<endl;      }        void add(int x, int y, int z)      {          a=x;b=y;c=z;          cout<<"addition of a+b+c is:"<<(a+b+c)<<endl;      }        void print()      {          cout<<"Class A's method is running"<<endl;      }  };    class B : public A  {      public:      void print()      {          cout<<"Class B's method is running"<<endl;      }  };    int main() {      A a1;        //method overloading (Compile-time polymorphism)      a1.add(6, 5);        //method overloading (Compile-time polymorphism)      a1.add(1, 2, 3);       B b1;       //Method overriding (Run-time polymorphism)     b1.print();  } |   **Output:**  addition of a+b is:11  addition of a+b+c is:6  Class B's method is running | | | |