Page No. : .....
Date : ...../...../.....

ASSIGNMENT - 2

NAME : AMRITANSHU KESHARI
BRANCH : COMPUTER SCIENCE ENGG.
ROLL NUMBER : 19402060006
SUBJECT : OOPS
SESSION : 2019 - 22
SEMESTER : 3rd SEMESTER
COLLEGE : GIOVERNMENT POLYTECHNIC COLLEGE, ADITYAPUR

Inheritance

Inheritance is a process in which one object acquires all the properties and behaviors of its parent object automatically. In such way, you can reuse, extend or modify the attributes and behaviors which are defined in other class.

The class which inherits the members of another class is called derived class and the class whose members are inherited is called base class. The derived class is the specialized class for the base class.

When one object acquires all the properties and behaviours of parent object i.e. known as the concept of inheritance in Object Oriented Programming System. It provides code reusability. It is used to achieve runtime polymorphism. The capability of a class to derive properties and charact-
er

characteristics from another class is called inheritance. Inheritance is one of the most important features of object oriented programming. One class inherits or acquires the properties of another class. Inheritance provides the idea of reusability of code and each sub class defines only those features that are unique to it, rest of the features can be inherited from the parent class. For example: dog, cat, cow can be derived class of Animal Base class.

C++ supports five types of inheritance are as follows as:

- Single Inheritance
- Multiple Inheritance
- Multilevel Inheritance
- Hybrid Inheritance
- Hierarchical Inheritance

Types Of Inheritance

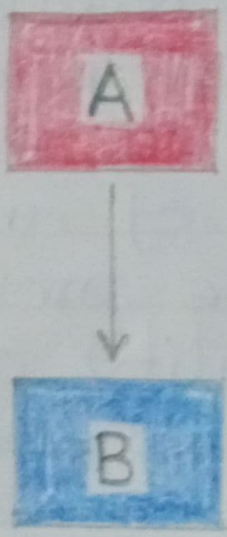
C++ Single Inheritance

Single inheritance can be defined as the inheritance in which a derived class is inherited from the only one base class. It is the most simplest form of inheritance. It provides reusability by allowing the derived class to inherit the features of the base class using objects. A class whose properties are inherited for re-usability is called parent class or super class or base class.

```
// C++ program of single inheritance

#include <iostream.h>
#include <conio.h>
class Amrit {
   public:
      int m, a;
   public:
      Amrit()
```

Single Inheritance



where 'A' is the base class, and 'B' is the derived class.

```
{
  cout << " Enter Mass :" <<endl ;
  cin >> m ;
  cout << " Enter Acceleration:" <<endl;
  cin >> a ;
   cout << " Force : " <<< m*a <<< endl;
  }
};

class Keshari : public Amrit
  {

  };

Void main ()
   {
      clrscr ();
      Keshari gm1 ;
      getch ();
   }
```

```
Enter Mass:
45
Enter Acceleration:
5

Force: 225
```

C++ Multiple Inheritance

Multiple Inheritance is a feature of C++ where a class can inherit from more than one classes. The constructors of inherited classes are called in the same order in which they are inherited. Multiple inheritance is the process of deriving a new class that inherits the attributes from two or more classes. In this type of inheritance a single derived class may inherit from two or more than two base classes that is one sub class is inherited from more than one base classes. Here, the number of base classes will be separated by a comma (',') and access mode for every base class must be specified.

```
// C++ program of multiple inheritance

# include <iostream.h>
# include <conio.h>
```

```
class Amrit {
    public :
      int m, a ;
    public :
      Amrit ()
       {
        cout << "Enter Mass :" << endl ;
        cin >> m ;
        cout << " Enter Acceleration :" << endl ;
        cin >> a ;
        cout << " Force : " << m*a << endl ;
       }
    };

class Anshu {
    public :
      float f, d ;
    public :
      Anshu ()
        {
          cout << endl << endl << endl ;
          cout << "Enter Force :" << endl ;
          cin >> f ;
          cout << " Enter Displacement :" <<
          endl ;
          cin >> d ;
```

Enter Mass :
7
Enter Acceleration :
11
Force : 77

Enter Force :
5
Enter Displacement :
21
Workdone : 105

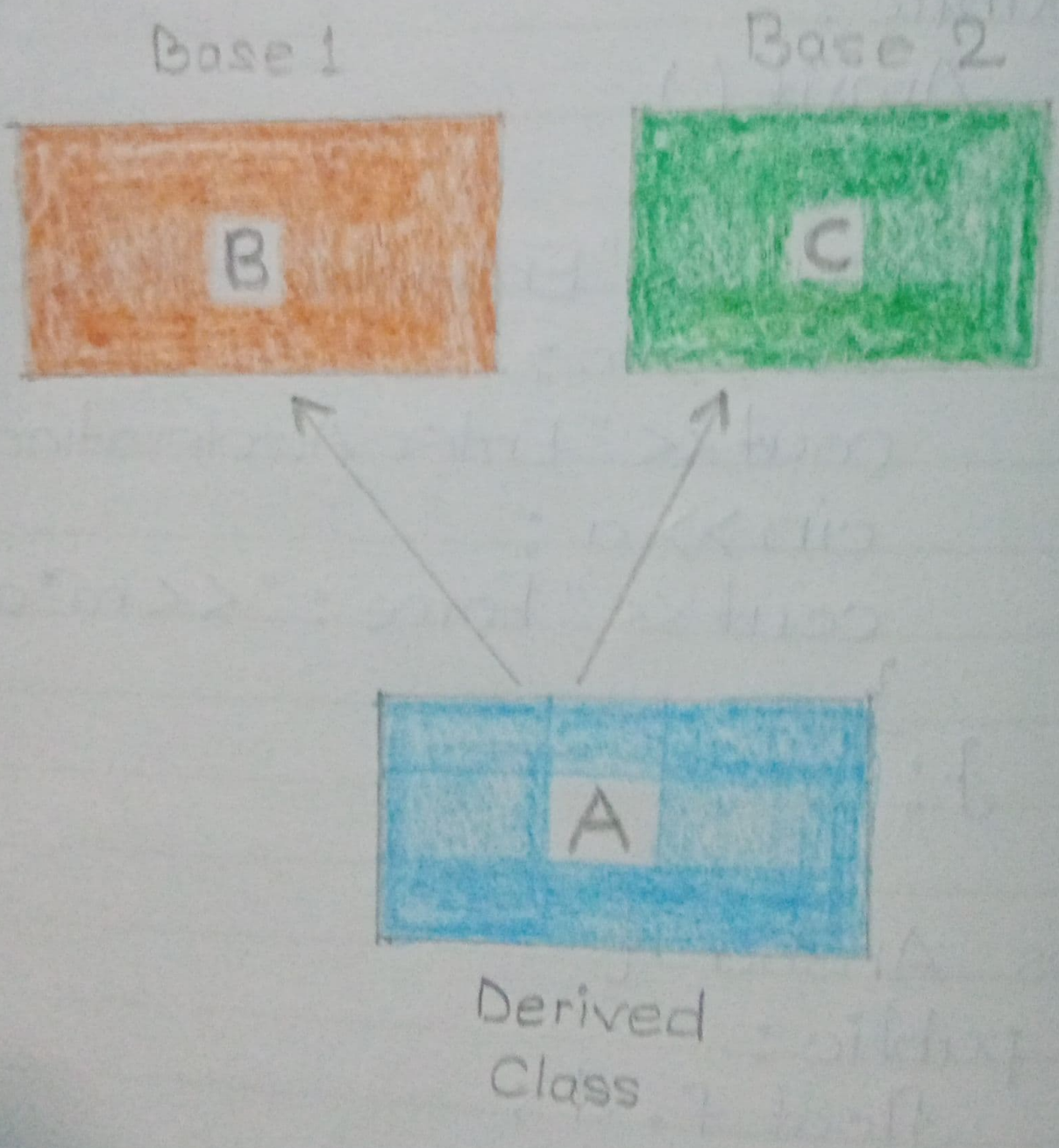
```
    cout << "Workdone : " << f*d << endl;
  }
};

class Keshari : public Amrit, public Anshu
  {

  };

void main()
  {
    clrscr();
    Keshari gm1;
    getch();
  }
```

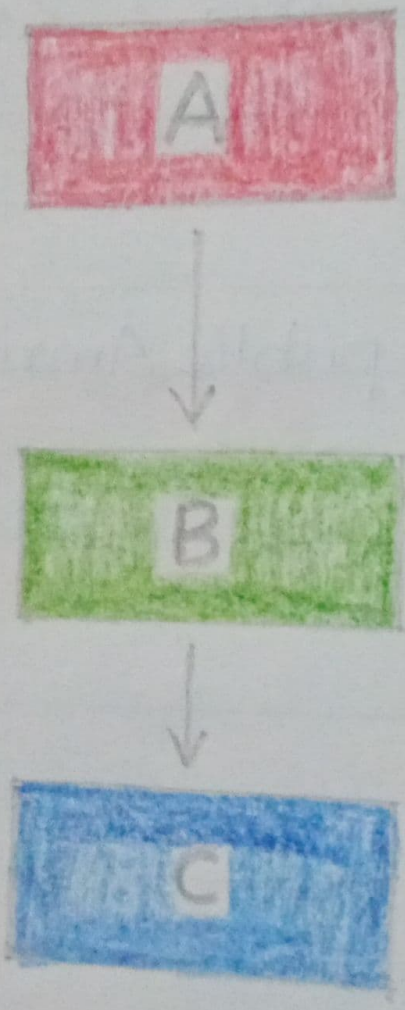
Multiple Inheritance



C++ Multilevel Inheritance

In C++ programming, not only you can derive a class from the base class but you can also derive a class from the derived class. This form of inheritance is known as multilevel inheritance. Here, class B is derived from the base class A and the class C is derived from the derived class B. A class can be derived from one class, which is already derived from the another class is called multilevel inheritance in Object Oriented Programming System. For example, if we take animals as a base class then mammals are the derived class which has features of animals and then humans are the also derived class that is derived from sub-class mammals which inherit all the features of mammals. As in other inheritance, based on the visibility mode used or access specifier used while deriving, the

Multilevel Inheritance



Base class

Intermediary class

Derived class

properties of the base class are derived. Access specifier can be private, protected or public.

```
// C++ program of multilevel
// inheritance

#include <iostream.h>
#include <conio.h>

class Amrit {
    public:
        int m, a;
    public:
        Amrit()
        {
            cout << "Enter Mass:" << endl;
            cin >> m;
            cout << "Enter Acceleration:" <<
            endl;
            cin >> a;
            cout << "Force :" << m*a
            << endl;
        }
};
```

```
class Anshu : public Amrit
  {

  };

class Keshari : public Anshu
  {

  };

Void main()
  {
    clrscr();
    Keshari gmt;
    getch();
  }
```

OUTPUT

```
Enter Mass:
 51
Enter Acceleration:
 11
Force : 561
```

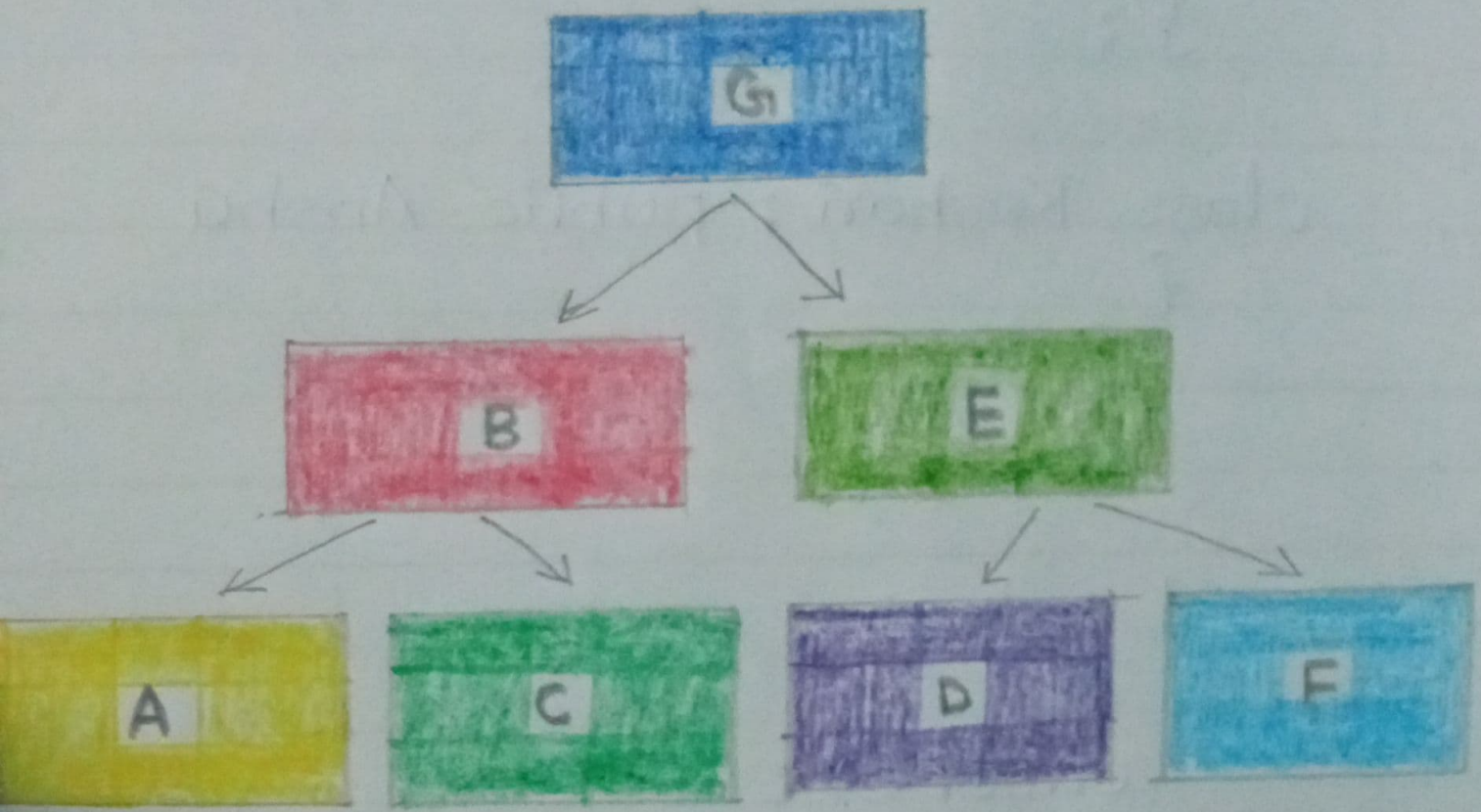
Hierarchical Inheritance

In this type of inheritance, more than one sub class is inherited from a single base class that is more than one derived class is created from a single base class. If more than one class is inherited from the base class, it's known as hierarchical inheritance. In hierarchical inheritance, all features that are common in child classes are included in the base class. Hierarchical inheritance is a kind of inheritance where more than one class is inherited from a single parent or base class. Especially those features which are common in the parent class is also common with the base class.

```
// C++ program of hierarchical
// Inheritance

# include <iostream.h>
# include <conio.h>
```

Hierarchical Inheritance



```
class Amrit {
    public:
      int m, a;
    public:
      Amrit()
       {
        cout << " Enter Mass : " << endl;
        cin >> m;
        cout << " Enter Acceleration : " <<
        endl;
        cin >> a;
        cout << " Force : " << m*a << endl;
       }
    };

class Anshu : public Amrit
    {

    };

class Keshari : public Amrit
    {

    };
```

OUTPUT

```
Enter Mass:
4
Enter Acceleration:
4
Force : 16
```

```
void main()
 {
    clrscr();
    Keshari gml;
    getch();
 }
```

This is the code of hierarchical inheritance in Object Oriented Programming System.

For example, a child inherits the traits of his/her parents. With inheritance, we can reuse the fields and methods of the existing class. Just like a car is a common class from which Audi, Ferrari, Maruti etc can be derived from class car.

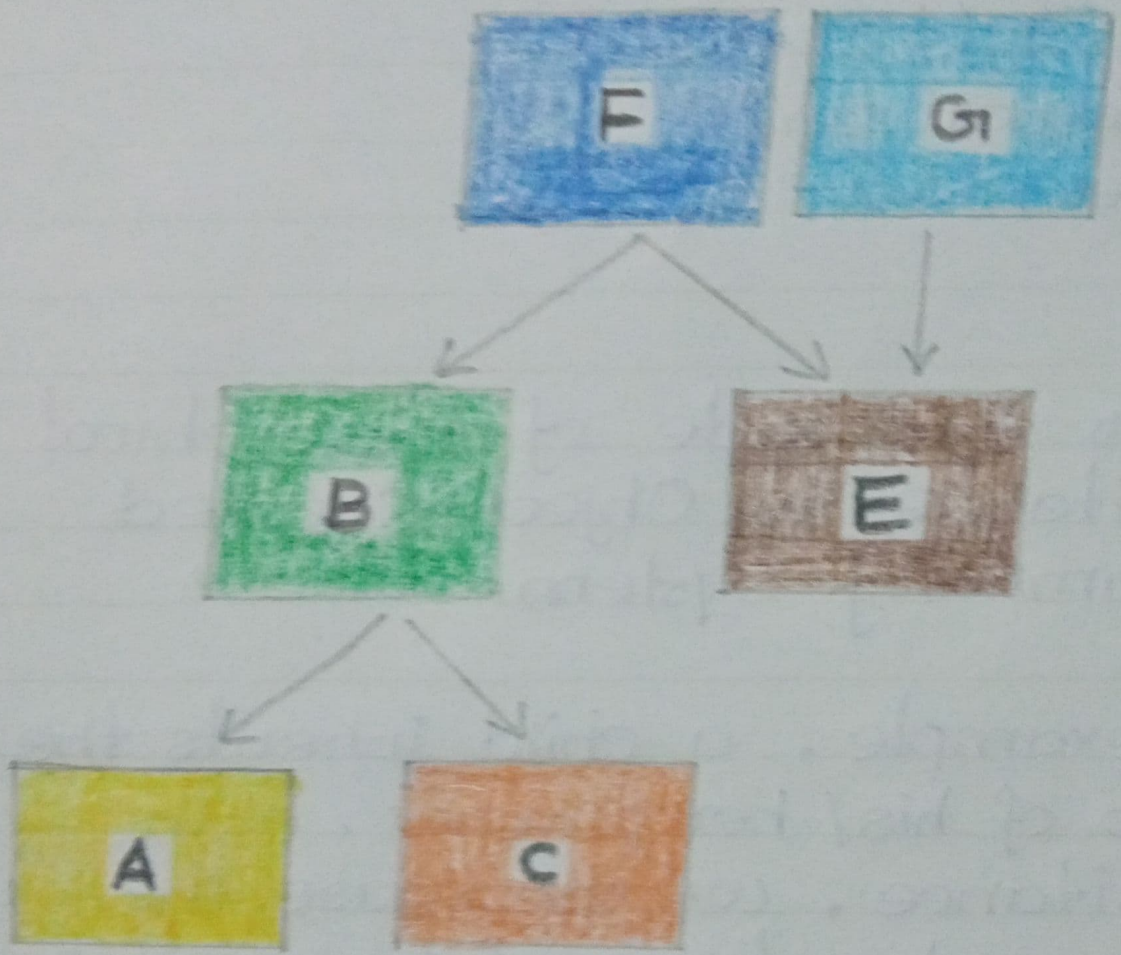
Hybrid Inheritance

The inheritance in which the derivation of a class involves more than one form of any inheritance is called hybrid inheritance. Basically C++ hybrid inheritance is combination of two or more types of inheritance. It can also be called multi path inheritance. Hybrid Inheritance is implemented by combining more than one type of inheritance. For example: Combining Hierarchical inheritance and multiple inheritance. Below image shows the combination of hierarchical and multiple inheritance.

```
// C++ program of hybrid inheritance

#include <iostream.h>
#include <conio.h>
```

Hybrid Inheritance



```
class Amrit {
    public:
     int m, a;
    public:
     Amrit()
      {
        cout << "Enter Mass : " << endl;
        cin >> m;
        cout << "Enter Acceleration : " << endl;
        cin >> a;
        cout << " Force : " << m*a << endl;
      }
    };

class Anshu {
    public:
     float f, d;
    public:
     Anshu()
      {
        cout << "Enter Force : " << endl;
        cin >> f;
        cout << "Enter Displacement : " << endl;
        cin >> d;
        cout << "Workdone : " << f*d << endl;
      }
    };
```

OUTPUT

```
Enter Mass :
5
Enter Acceleration :
7

Force : 35


Enter Force :
2
Enter Displacement
7
Workdone : 14
```

```
class Keshari : public Amrit
  {

  };

class physics : public Amrit, public Anshu
  {

  };

void main()
  {
    clrscr();
    physics gm1;
    getch();
  }
```