Pure Virtual Function

Definition

Pure virtual functions are used -

- If a function doesn't have any use in the base class but the function must be implemented by all its derived classes.
- For example suppose, we have derived Triangle, Square and Circle classes from the Shape class, and we want to calculate the area of all these shapes.
- In this case, we can create a pure virtual function named calculateArea() in the Shape class.

Example

- A pure virtual function doesn't have the function body and it must end with 0. For example,

```
class shape {
public:
virtual void calculateArea()=0;
}
```

Abstract Class

- A class that contains a pure virtual function is known as an abstract class. In the above example, the class Shape is an abstract class.
- We cannot create objects of an abstract class. however, we can derive classes from them, and use their data members and member functions (except pure virtual functions).

Example

```
#include <iostream>
 using namespace std;
// Abstract class
class Shape {
protected:
float dimension;
public:
void getDimension() {
cin >> dimension;
}
virtual float calculateArea() = 0;
 };
class Square : public Shape {
public:
float calculateArea() {
 return dimension * dimension;
} };
class Circle : public Shape {
public: float calculateArea() {
 return 3.14 * dimension * dimension;
 } };
```

Continue..

```
int main()
{
Square square;
Circle circle;
cout << "Enter the length of the square: ";
square.getDimension();
cout << "Area of square: " << square.calculateArea() << endl;
cout << "\nEnter radius of the circle: ";
 circle.getDimension();
cout << "Area of circle: " << circle.calculateArea() << endl;
 return 0;
 }
```