Department of Computer Science & Engineering, NITS
CS212 - Object Oriented Programming Laboratory
Session : January - June 2024

Lab-4: Polymorphism

Polymorphism, means the ability to take more than one form. An operation may exibit different behaviours in different instances. The behaviour depends upon the types of data used in the operation.

The over loaded member *functions* are "selected" for invoking by matching arguments, both

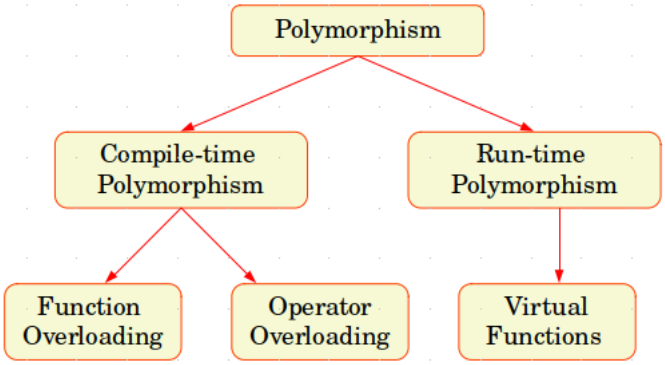

Figure 1: Polymorphism in Object Oriented Programming

type and numbers. This information is known to the compiler at the compile time and therefore, compiler is able to select the appropriate function for a particular call at the compile time itself. This is called **early binding** or **static binding** or **static linking** or **compile time polymorphism**.If the appropriate member function could be selected while the program is running, this is known as **run time polymorphism**. C++ supports a mechanism known as **virtual function** to achieve run time polymorphism (other terms used as, **late binding**, **dynamic binding**).

C++ allows the user to **overload** most of the **operators** so that the operators can work effectively in a specific application. It does not allow the user to create new operators. Most of the existing operators can be overloaded to manipulate class objects.

Assignments :

1. Write a program to illustate how to overload the operators + , * , == , != , » , and « . In this exercise, first redefine the class rectangleType by declaring the instance variables as protected and then overload additional operators as defined in parts a to c.

   (a) Overload the pre- and post-increment and decrement operators to increment and decrement, respectively, the length and width of a rectangle by one unit. (Note that after decrementing the length and width, they must be postive.)

   (b) Overload the binary operator - to subtract the dimensions of one rectangle from the corresponding dimensions of another rectangle. If the resulting dimensions are not positive, output an appropriate message and do not perform the operation.

   (c) The operators == and != are overloaded by considering the lengths and widths of rectangles. Redefine the functions to overload the relational operator by considering the areas of rectangles as follows: Two rectan- gles are the same, if they have the same area; otherwise, the rectangles are not the same. Similary, rectangle yard1 is greater than rectangle yard2 if the area of yard1 is greater than the area of yard2 . Overload the remaining relational operators using similar definitions.

(d) Write the definitions of the functions to overload the operators defined in parts a to c.

2. Design a class to perform various matrix operations. A matrix is a set of numbers arranged in rows and columns. Therefore, every element of a matrix has a row position and a column position. If A is a matrix of five rows and six columns, we say that the matrix A is of the size 5 × 6 and sometimes denote it as A 5 × 6 . Clearly, a convenient place to store a matrix is in a two-dimensional array. Two matrices can be added and subtracted if they have the same size. Suppose A = $[a_{ij}$] and B = $[b_{ij}]$ are two matrices of the size m × n, in which a_{ij} denotes the element of A in the ith row and the jth column, and so on. The sum and difference of A and B are given by:
$A + B = [a_{ij} + b_{ij}]$
$A - B = [a_{ij} - b_{ij}]$
The multiplication of A and B (A * B) is defined only if the number of columns of A is the same as the number of rows of B. If A is of the size m × n and B is of the size n × t, then A * B = $[c_{ik}$] is of the size m × t.
Design and implement a class matrixType that can store a matrix of any size. Overload the operators + , - , and * to perform the addition, subtraction, and multiplication operations, respectively, and overload the operator « to output a matrix. Also, write a test program to test various operations on the matrices.

3. Answer the following questions.

(a) Name the operators that cannot be overloaded.

(b) What is the difference between the two statements return this; and return *this; ?

(c) What is the difference between a friend function of a class and a member function of a class?

(d) Which operators must be overloaded as members of a class?

(e) What is a virtual function and why do we need it?

(f) Explain, with an example, how you would create space for an array of objects using pointers.

(g) What does **this** pointer point to?

(h) What are the applications of **this** pointer?

Submission Details :

- Date of Issue : 20 - 02 - 2024
- Date of Submission : (Soft Copy, before 2 PM, 27-02-2024)
- Evaluation time : 2 - 5 PM