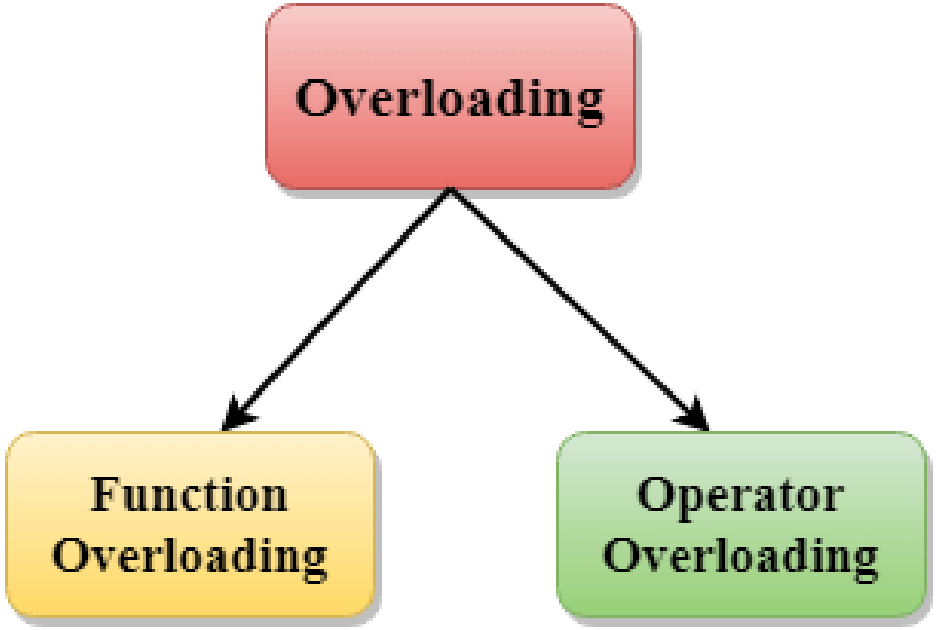
CAP444
OBJECT ORIENTED PROGRAMMING USING C++

Unit2

Created By:
Kumar Vishal
(SCA), LPU

Polymorphism



❖ Functions overloading

- same function name but different parameters.
- same function name with different signature
- example of polymorphism(**compile time)**
- overloaded functions should be there in same scope.

What will be output?

```
#include <iostream>
   using namespace std;
   void print(int i)
   {
      cout << i;
   }
   void print(double  f)
   {
      cout << f;
   }
   int main()
   {
      print(5);
      print(500.263);
      return 0;
   }
```

A) 5500.263

B) 500.2635

C) 500.263

```
#include <iostream>
 using namespace std;
class Teacher
 {
    public:
    string name;
     Teacher()
     {
        name="kumar";
      }
     Teacher(string str1)
     {
        name=str1;
     }
   void getName()   {
      cout<<name;   }
```

```
int main()
 {
  Teacher T1("vishal"),T2("Ajay"),T3;
  T3=T1+T2;
  T3.getName();
  return 0;
 }
```

Operator overloading

Operator overloading is a compile-time polymorphism in which the operator is overloaded to provide the special meaning to the user-defined data type.

- You can redefine built in operators except few:
  - Scope operator (::)
  - Sizeof
  - member selector(.)
  - member pointer selector(.*)
  - ternary operator(?:)

Operators which can overload

+	-	*	/	%	^
&	|	~	!	,	=
<	>	<=	>=	++	--
<<	>>	==	!=	&&	||
+=	-=	/=	%=	^=	&=
|=	*=	<<=	>>=	[]	()
->	->*	new	new []	delete	delete []

Operator Overloading Syntax:

Return_type operator operator_Symbol(parameters)

{

}

Example:

```
Teacher operator+(Teacher &t)
   {
      return name+t.name;
    }
```

- Unary operators operate on only one operand

Ex:

i++

- Binary operators work on two operands

Ex:

+

Which of the following operator cannot be overloaded?

a) +

b) ?:

c) –

d) %

What is a binary operator?
a) Operator that performs its action on a single operand
b) Operator that performs its action on two operand
c) Operator that performs its action on three operand
d) Operator that performs its action on any number of operands

Operator overloading for Unary operators:

The unary operators operate on a single operand :

- The increment (++) and decrement (--) operators.
- The unary minus (-) operator.
- The logical not (!) operator.

Rules for Operator Overloading

- Existing operators can only be overloaded.
- The overloaded operator contains at least one operand of the user-defined data type.
- When unary operators are overloaded through a member function take no explicit arguments, but, if they are overloaded by a friend function, takes one argument.
- When binary operators are overloaded through a member function takes one explicit argument, and if they are overloaded through a friend function takes two explicit arguments.

Friend function

- It can access all private and protected member of a class
- It can be call without object of the class
- It can define out side of the class scope

Rule:

Prototypes of friend function must be declare inside the class

It can be declared either in the private or the public part.

Simple example : friend function

```
#include <iostream>
using namespace std;
class A
{
private:
   int x;
public:
   A()
   {
      x=10;
   }
private:
   friend void newfriend(A &a);
};
void newfriend(A &a)
{
   a.x=20;
   cout<<a.x;
}
int main()
{
A a1;
newfriend(a1);
return 0;
}
```

Overloading binary operators using friend function

Friend function takes two parameters in case when we want to overload binary operators using friend function

Ex:

friend A operator +(A &x, A &y);

Example:


LOVELY PROFESSIONAL UNIVERSITY
PUNJAB (INDIA)
L
P
U


LOVELY PROFESSIONAL UNIVERSITY
PUNJAB (INDIA)
L
P
U

Any Query?