Question No: 27 (Marks: 2)
How many arguments a Unary Operator take? Can we make a binary operator as unary operator?
Answer:-
Unary operator takes only one arguments like i++ or i-- (Post increment or post decrement operators for integers) or ++i,--i (Pre increment or pre decrement operators for integers) ,we can not make Unary operator as binary or binary as Unary operator.

Question No: 28 (Marks: 2)
Which arithmetic operators cannot have a floating point operand?
Answer:-
Modulus operator
This operator can only be used with integer operands ONLY

Question No: 29 (Marks: 2)
What are manipulators? Give one example.

Answer:-
The manipulators are like something that can be inserted into stream, effecting a change in the behavior. For example, if we have a floating point number, say pi (л), and have written it as *float pi = 3.1415926* ; Now there is need of printing the value of pi up to two decimal places i.e. 3.14 . This is a formatting functionality. For this, we have a *manipulator* that tells about width and number of decimal points of a number being printed.
OR
Answer: Manipulators are operators used in C++ for formatting output. The data is manipulated by the programmer's choice of displayed.
Endl manipulator: This manipulator has the same functionality as the „\n newline character.

Question No: 30 (Marks: 2)
Write down piece of code that will declare a matrix of 3x3. And initialize all its locations with 0;

Answer:-
```
int matrix [3] [3] ;

matrix [0] [0] = 0;
matrix [0] [1] = 0;
matrix [0] [2] = 0;
matrix [1] [0] = 0;
matrix [1] [2] = 0;
matrix [1] [2] = 0;
matrix [2] [0] = 0;
matrix [2] [1] = 0;
```

matrix [2] [2] = 0;

Another answer:-
we can also do it as given below
int matrix [3][3] = { 0 }; //all elements 0

Question No: 31 (Marks: 3)
Which one (copy constructor or assignment operator) will be called in each of the following code segment?
1) Matrix m1 (m2);
2) Matrix m1, m2;
m1 = m2;
3) Matrix m1 = m2;

Answer:-
1) Matrix m1 (m2); **copy constructor**
2) Matrix m1, m2;
m1 = m2; **assignment operator**
3) Matrix m1 = m2; **assignment operator**

Question No: 32 (Marks: 3)
What will be the output of following function if we call this function by passing int 5?

```
template <class T>
T reciprocal(T x)
{
return (1/x);
}
```

Answer:-
0
The output will zero as 1/5 and its .05 but conversion to int make it zero
Above is prototype of template class so assume passing an int and returning an int

Question No: 33 (Marks: 3)

Identify the errors in the following member operator function and also correct them.

```
math * operator(math m);

math * operator (math m)
{
    math temp;
    temp.number= number * number;
    return number;

}
```

Answer:-

The errors are in the arguments of the member operation function and also in the body of operator member function.

Correct function should be

```
math *operator(math *m);

math *operator (math *m)
{
    math temp;
    temp = m;
    temp.number= number * number;
    return temp.number;

}
```

Question No: 34 (Marks: 5)
Write a program which defines three variables of type double which store three different values including decimal points, using setprecision manipulators to print all these values with different number of digits after the decimal number.

Answer:-

```
#include <iostream>
#include <iomanip>

int main ()
{
  double x1 = 12345624.72345
    double x2 =  987654.12345
    double x3 =  1985.23456
  cout << setprecision (3) << x1<< endl;
  cout << setprecision (4) << x2 << endl;
    cout << setprecision (5) << x3<< endl;

  return 0;
}
```

Question No: 35 (Marks: 5)
What are the advantages and disadvantages of using templates?
Answer:-Page 518
Many thing can be possible without using templates but it do offer several clear advantages not offered by any other techniques:
Advanatages:
• Templates are easier to write than writing several versions of your similar code for different types. You

create only one generic version of your class or function instead of manually creating specializations.
• Templates are type-safe. This is because the types that templates act upon are known at compile time, so the compiler can perform type checking before errors occur.
• Templates can be easier to understand, since they can provide a straightforward way of abstracting type information.
• It help in utilizing compiler optimizations to the extreme. Then of course there is room for misuse of the templates. On one hand they provide an excellent mechanism to create specific type-safe classes from a generic definition with little overhead.

Disadvantages:

On the other hand, if misused
• Templates can make code difficult to read and follow depending upon coding style.
• They can present seriously confusing syntactical problems esp. when the code is large and spread over several header and source files.
• Then, there are times, when templates can "excellently" produce nearly meaningless compiler errors thus requiring extra care to enforce syntactical and other design constraints. A common mistake is the angle bracket problem.

Question No: 36 (Marks: 5)

Suppose a program has a math class having only one data member **number**.
Write the declaration and definition of operator function to overload + operator for the statements of main function.

```
math obj1, obj2;
obj2= 10 + obj1  ;
```

Answer:-

```
#include <iostream.h>
math
{
mth operator + (obj1,obj2)
mth operator + (obj1,obj2)
{
 mth operator + (obj1,obj2)
mth operator + (obj1,obj2)
}
}
```

Question No: 32 (Marks: 3)
Is it possible to define two functions as given below? Justify your answer.
func(int x, int y)
func(int &x, int &y)
Answer:-
No, we cannot define two functions as func(intx, inty) func(int &x, int&y) because it's give an error function not initializing.

Question No: 33 (Marks: 3)
What happens when we use new and delete operator?
Answer:
When we use ***new*** operator to create objects the memory space is allocated for the object and then its constructor is called. Similarly, when we use ***delete*** operator with our objects, the destructor is called for the object before deallocating the storage to the object.

Question No: 34 (Marks: 5)
What is the difference between function overloading and operator overloading?
Answer:-
Difference b/w function overloading and operator overloading is:
In function overloading, the functions have the same name but differ either by the number of arguments or the type of the arguments.
Operator overloading is to allow the same operator to be bound to more than one implementation, depending on the types of the operands.

Question No: 35 (Marks: 5)
Why the first parameter of operator function for << operator must be passed by reference?
Answer:-
Operator<<'s first parameter must be an ostream passed by reference. Its second parameter, the IntList that is printed, does not have to be passed as a const-reference parameter; however it is more efficient to pass it by reference than by value (since that avoids a call to the copy constructor), and it should not be modified by operator<<, so it should be a const reference parameter

Question No: 36 (Marks: 5)
Read the given below code and explain what task is being performed by this function

```
Matrix :: Matrix ( int row , int col )
{
   numRows = row ;
   numCols = col ;
   elements = new ( double * ) [ numRows ] ;
   for ( int  i = 0 ; i < numRows ; i ++ )
         {
      elements [ i ] = new double [ numCols ] ;
      for ( int j = 0 ; j < numCols ; j ++ )
            elements [ i ] [ j ] = 0.0 ;
    }
}
```

Hint : This function belong to a matrix class, having
Number of Rows = numRows

Number of Columns = numCols

Answer:
In this code the matrix function is defined, it get the number of rows from the user and create the row of matrix and then get the columns from the user and create the columns. The New is showing for creating more array space for the data which user enters. The elements [i][j] will print the data in matrix form.
http://vustudents.ning.com

Question No: 27 (Marks: 2)
What is the difference between **switch** statement and **if** statement.
Answer:
1.if statement is used when we have to check two conditions while switch is a multi conditional control statement
2. SWITCH statement can be executed with all cases if the "break" statement is not used whereas IF statement has to be true to be executed further.

Question No: 28 (Marks: 2)
How can we initialize data members of contained object at construction time?
Answer:
Initializer list is used to initialize the contained objects at the construction time.

Question No: 29 (Marks: 2)
How the data members of a class are initialized with meaningful values?
Answer: Page 334
The C++ compiler generates a default constructor for a class if the programmer does not provide it. But the default constructor does not perform any data members initialization. Therefore, it is good practice that whenever you write a class, use a constructor function to initialize the data members to some meaningful values.

Question No: 30 (Marks: 2)
Can we overload ***new*** and ***delete*** operators?
Answer: Page 412
Yes, it is possible to overload *new* and *delete* operators to customize memory management. These operators can be overloaded in global (non-member) scope and in class scope as member operators.

Question No: 31 (Marks: 3)
What will be the output of following functions if we call these functions three times?

1)
```
void func1(){
int x = 0;
x++;
cout << x << endl;
}
```
Answer:
1
1

1

2)

```
void func2(){
static int x = 0 ;
x++;
cout << x << endl ;
}
```

Answer:

1
2
3

Question No: 32 (Marks: 3)

What is the keyword '**this**' and what are the uses of '**this**' pointer?

Answer:

'this' is use to refer the current class member without using the name of the class.We cannot use it as a variable name. 'this' pointer is present in the function, referring to the calling object. this pointer points to the current object.

Question No: 33 (Marks: 3)

Suppose an object of class A is declared as data member of class B.

(i) The constructor of which class will be called first?

Answer: A

(ii) The destructor of which class will be called first?

Answer: B

Question No: 34 (Marks: 5)

Write the general syntax of a class that has one function as a friend of a class along with definition of friend function.

Answer:

```
class frinedclass{
public:
friend int compute(exforsys e1)
};
Int compute(exforsys e1)
{
//Friend Function Definition which has access to private data
return int(e1.a+e2.b)-5;
}
```

Question No: 35 (Marks: 5)
Write down the disadvantages of the templates.
Answer: Repeated

Question No: 36 (Marks: 5)
Write a program which defines five variables which store the salaries of five employees, using setw and setfill manipulators to display all these salaries in a column.

Note: Display all data with in a particular width and the empty space should be filled with character x
Output should be displayed as given below:
xxxxxx1000
xxxxxx1500
xxxxx20000
xxxxx30000
xxxxx60000
Answer:

```
#include <iostream.h>
#include <iomanip.h>
main(){
int sal1 =1000;
int sal2 =1500;
int sal3 =20000;
int sal4 =30000;
int sal5 =60000;
cout << setfill ('x') << setw (10);
cout<< sal1<<endl;
cout << setfill ('x') << setw (10);
cout<< sal2<<endl;
cout << setfill ('x') << setw (10);
cout<< sal3<<endl;
cout << setfill ('x') << setw (10);
cout<< sal4<<endl;
cout << setfill ('x') << setw (10);
cout<< sal5<<endl;
int i=0;
cin>>i; // to stop the screen to show the output
}
```