Assignment - 2

1. A) List out the categories of function overloading. 2

Answer:-

* 1. Which OOPs concept is being implemented in function overloading.

Answer:- polymorphism

1. What are static data members. How are they different from instance variables? Also mention, can we access non static members through static member functions?

Answer:- Static data members are **class members that are declared using static keywords**. Instance variable are declared without any keyword and return type but static member functuion includes keyword static and return type.

No we can not access non static members through a static member function . but we can access non static member and static member through a non static member function.

1. Differentiate between macro and inline functions

Answer:-

| S.NO | Inline | Macro |
| --- | --- | --- |
| 1. | An inline function is defined by the **inline** keyword. | Whereas the macros are defined by the **#define** keyword. |
| 2. | Through inline function, the class’s data members can be accessed. | Whereas macro can’t access the class’s data members. |
| 3. | In the case of inline function, the program can be easily debugged. | Whereas in the case of macros, the program can’t be easily debugged. |
| 4. | In the case of inline, the arguments are evaluated only once. | Whereas in the case of macro, the arguments are evaluated every time whenever macro is used in the program. |
| 5. | In C++, inline may be defined either inside the class or outside the class. | Whereas the macro is all the time defined at the beginning of the program. |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

1. What is this pointer? Why it is needed. Explain any two applications of this pointer with corresponding code segment.

Answer:- ‘this' is one of the keywords in programming.

The 'this' pointer points to the object itself. The compiler itself uses a 'this' pointer when the user wants some data of a particular object.

Example,

//data is a private data member

Int max(xyz obj)

{

If(this->data>obj.data)

Return this->data;//points to ‘data' of obj1

Else

Return obj->data;

}

The function call being

Int y=Obj1.max(obj2);

1. A) Write a C++ class to calculate simple interest and compound interest. Members are to be accessed using member dereferencing operators/ pointer to member operators.

Answer:- #include <iostream.h>

#include <math.h>

int main()

{

int p;

float r;

int t;

cout<<"Enter principal amount: ";

cin>>p;

cout<<"Enter rate of interest: ";

cin>>r;

cout<<"Enter time period (in years): ";

cin>>t;

float si = p\*r\*t;

cout<<"Simple interest is: Rs."<<si<<endl;

float total = p \* pow((1+r/12), (12\*t));

float ci = total - p;

cout<<"Compound interest is: Rs. "<<ci<<endl;

return 0;

}

* 1. Write a C++ program using classes to perform addition of time in hours and minutes.

Answer:- /\*C++ program to create class to read and add two times.\*/

#include <iostream>

using namespace std;

class Time

{

private:

int hours;

int minutes;

int seconds;

public:

void getTime(void);

void putTime(void);

void addTime(Time T1,Time T2);

};

void Time::getTime(void)

{

cout << "Enter time:" << endl;

cout << "Hours? "; cin>>hours;

cout << "Minutes? "; cin>>minutes;

cout << "Seconds? "; cin>>seconds;

}

void Time::putTime(void)

{

cout << endl;

cout << "Time after add: ";

cout << hours << ":" << minutes << ":" << seconds << endl;

}

void Time::addTime(Time T1,Time T2)

{

this->seconds=T1.seconds+T2.seconds;

this->minutes=T1.minutes+T2.minutes + this->seconds/60;;

this->hours= T1.hours+T2.hours + (this->minutes/60);

this->minutes %=60;

this->seconds %=60;

}

int main()

{

Time T1,T2,T3;

T1.getTime();

T2.getTime();

//add two times

T3.addTime(T1,T2);

T3.putTime();

return 0;

}

1. A) Write a C++ program using classes to add two complex numbers together.
2. Answer:- // C++ program to Add two complex numbers
3. #include <bits/stdc++.h>
4. **using** **namespace** std;
6. **class** Complex {
7. **public**:
8. **int** real; // To store real part of complex number
9. **int** imaginary; // To store imaginary part of complex number
11. Complex()
12. {
13. // Initial values are zero
14. real = 0;
15. imaginary = 0;
16. }
17. Complex(**int** r, **int** i)
18. {
19. real = r; // r is initialized during object creation
20. imaginary = i; // i is initialized during object creation
21. }
23. Complex addComplexNumber(Complex C1, Complex C2)
24. {
26. Complex res; // result object of complex class
28. // adding real part of complex numbers
29. res.real = C1.real + C2.real;
31. // adding Imaginary part of complex numbers
32. res.imaginary = C1.imaginary + C2.imaginary;
34. // returning the sum
35. **return** res;
36. }
37. };
39. // Main Class
40. **int** main()
41. {
43. // First Complex number
44. Complex C1(4, 5);
46. // printing first complex number
47. cout << "Complex number 1 : " << C1.real
48. << " + i" << C1.imaginary << endl;
50. // Second Complex number
51. Complex C2(8, 9);
53. // printing second complex number
54. cout << "Complex number 2 : " << C2.real
55. << " + i" << C2.imaginary << endl;
57. // for Storing the sum
58. Complex C3;
60. // calling addComplexNumber() method
61. C3 = C3.addComplexNumber(C1, C2);
63. // printing the sum
64. cout << "Sum of complex number : "
65. << C3.real << " + i"
66. << C3.imaginary;
68. cout << endl
69. << endl;
70. // Test for second input
71. // First Complex number
72. Complex A(2, 7);
74. // printing first complex number
75. cout << "Complex number 1 : " << A.real
76. << " + i" << A.imaginary << endl;
78. // Second Complex number
79. Complex B(10, 6);
81. // printing second complex number
82. cout << "Complex number 2 : " << B.real
83. << " + i" << B.imaginary << endl;
85. // for Storing the sum
86. Complex C;
88. // calling addComplexNumber() method
89. C = C.addComplexNumber(A, B);
91. // printing the sum
92. cout << "Sum of complex number : "
93. << C.real << " + i"
94. << C.imaginary;
95. }
    1. Write a C++ program using classes to overload the method area(). The prototypes are as under

int area(int)  Area of square

int area(int,int)  Area of 41][ḍ©\_)O rectange int area(int,int, int)  Area of triangle

1. A) Determine the output, explain your answer: 1.5

#include <iostream> using namespace std; class A

{

int x; A \*p; public:

void setdata(int d, A \*p1)

{

x = d; p = p1;

}

void display()

{

cout<<x<<" "<<(\*p).x<<endl;

}

};

int main(){

A a, b; a.setdata(10, &b); b.setdata(20, &a); a.display();

b.display(); return 0;

}

1. Write a C++ program using class to read a matrix and transpose it. Use the concept of array of objects. 2.5

Code:- #include <iostream>

using namespace std;

int main() {

int a[10][10], transpose[10][10], row, column, i, j;

cout << "Enter rows and columns of matrix: ";

cin >> row >> column;

cout << "\nEnter elements of matrix: " << endl;

// Storing matrix elements

for (int i = 0; i < row; ++i) {

for (int j = 0; j < column; ++j) {

cout << "Enter element a" << i + 1 << j + 1 << ": ";

cin >> a[i][j];

}

}

// Printing the a matrix

cout << "\nEntered Matrix: " << endl;

for (int i = 0; i < row; ++i) {

for (int j = 0; j < column; ++j) {

cout << " " << a[i][j];

if (j == column - 1)

cout << endl << endl;

}

}

// Computing transpose of the matrix

for (int i = 0; i < row; ++i)

for (int j = 0; j < column; ++j) {

transpose[j][i] = a[i][j];

}

// Printing the transpose

cout << "\nTranspose of Matrix: " << endl;

for (int i = 0; i < column; ++i)

for (int j = 0; j < row; ++j) {

cout << " " << transpose[i][j];

if (j == row - 1)

cout << endl << endl;

}

return 0;

}

1. Determine the error in the following program 1

#include<iostream> using namespace std; class addition

{

int p,q,r,s;

void input(void)

{

cout<<"Enter three nos.";

cin>>p>>q>>r>>s;

}

void show()

{

cout<<"Addition is"<<add();

}

int add()

{

s = p+q+r; return (s);

}

};

int main()

{

addition x; x.input();

x.show(); return 0;

}

Answer:- we have initialized the variable S

To store the sum in it but we are adding s along with the other numbers.. hence the compiler will throw an error or we will get wrong output.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*