**SUBMITTED BY :**

**NAME : ZISHNENDU SARKER**

**ROLL NO.: 2K19/CO/450**

**SUBJECT : OOP LAB**

**LAB ASSIGNMENT NO. : 07**

**SUBMITTED TO :**

**DIKSHA RUHELA MAM**

**QUESTION 1:** Create a class complex with real and imaginary parts as member variables, member function get () and disp () to input and display a complex number respectively. Write a program using the above class to overload + and – operators to perform addition and subtraction of two complex numbers.

**ANS:**

**#include <iostream>**

**using namespace std;**

**class Complex**

**{**

**private:**

**float real;**

**float imag;**

**public:**

**Complex(): real(0), imag(0){ }**

**void input()**

**{**

**cout << "Enter real and imaginary parts respectively: ";**

**cin >> real;**

**cin >> imag;**

**}**

**// Operator overloading**

**Complex operator - (Complex c2)**

**{**

**Complex temp;**

**temp.real = real - c2.real;**

**temp.imag = imag - c2.imag;**

**return temp;**

**}**

**void output()**

**{**

**if(imag < 0)**

**cout << "Output Complex number: "<< real << imag << "i";**

**else**

**cout << "Output Complex number: " << real << "+" << imag << "i";**

**}**

**};**

**int main()**

**{**

**Complex c1, c2, result;**

**cout<<"Enter first complex number:\n";**

**c1.input();**

**cout<<"Enter second complex number:\n";**

**c2.input();**

**result = c1 - c2;**

**result.output();**

**return 0;**

**}**

**QUESTION 2:** Write a program in C++ to overload subscript [ ] operator.

ANS:

#include <iostream>

using namespace std;

const int SIZE = 10;

class safearay {

private:

int arr[SIZE];

public:

safearay() {

register int i;

for(i = 0; i < SIZE; i++) {

arr[i] = i;

}

}

int &operator[](int i) {

if( i > SIZE ) {

cout << "Index out of bounds" <<endl;

// return first element.

return arr[0];

}

return arr[i];

}

};

int main() {

safearay A;

cout << "Value of A[2] : " << A[2] <<endl;

cout << "Value of A[5] : " << A[5]<<endl;

cout << "Value of A[12] : " << A[12]<<endl;

return 0;

}

**QUESTION 4:** Write a program to compare two strings by overloading == operator.

ANS:

#include <iostream>

using namespace std;

#include <string.h>

class String{

private:

enum { SZ = 80 };

char str[SZ];

public:

String(){ strcpy(str, ""); }

String( char s[] ){ strcpy(str, s); }

void display() const{ cout << str; }

void getstr(){ cin.get(str, SZ); }

bool operator == (String ss) const{

return ( strcmp(str, ss.str)==0 ) ? true : false;

}

};

int main(){

String s1 = "yes";

String s2 = "no";

String s3;

cout << "\nEnter 'yes' or 'no': ";

s3.getstr();

if(s3==s1)

cout << "You typed yes\n";

else if(s3==s2)

cout << "You typed no\n";

else

cout << "You didn't follow instructions\n";

return 0;

}

**QUESTION 5:** Write a program to add two string using + operator overloading.

ANS:

#include<iostream>

#include<string.h>

using namespace std;

class String

{

public:

char str[20];

public:

void accept\_string()

{

cout<<"\n Enter String : ";

cin>>str;

}

void display\_string()

{

cout<<str;

}

String operator+(String x) //Concatenating String

{

String s;

strcat(str,x.str);

strcpy(s.str,str);

return s;

}

};

int main()

{

String str1, str2, str3;

str1.accept\_string();

str2.accept\_string();

cout<<"\n ----------------------------------------------";

cout<<"\n\n First String is : ";

str1.display\_string(); //Displaying First String

cout<<"\n\n Second String is : ";

str2.display\_string(); //Displaying Second String

cout<<"\n ----------------------------------------------";

str3=str1+str2; //String is concatenated. Overloaded '+' operator

cout<<"\n\n Concatenated String is : ";

str3.display\_string();

return 0;

}

**QUESTION 6:** Write a program to overload new and delete operator.

ANS:

#include <iostream>

#include <string>

class MyClass{

float\* p= new float[100];

};

class MyClass2{

int five= 5;

std::string s= "hello";

};

int main()

{

int\* myInt= new int(1998);

double\* myDouble= new double(3.14);

double\* myDoubleArray= new double[2]{1.1,1.2};

MyClass\* myClass= new MyClass;

MyClass2\* myClass2= new MyClass2;

delete myDouble;

delete [] myDoubleArray;

delete myClass;

delete myClass2;

// getInfo();

}

QUESTION 8 : Write a program to overload == operator to check whether two circles are equal or not. (Two circles are said to be equal if their radius is same and center has same coordinate)

ANS:

#include <iostream>

using namespace std;

class Circle {

private:

float radius;

public:

Circle(float r = 0) : radius(r) { }

void changeRadius(int radius) { this->radius = radius; }

float getRadius() { return radius; }

float getArea() const { return 3.14 \* radius \* radius; }

bool operator==(Circle a);

};

inline bool Circle::operator==(Circle a)

{

return this->radius == a.radius;

}

int main()

{

Circle A(10), B(20), C(10);

cout << "Circle A : Radius = " << A.getRadius() << " units, Area = "

<< A.getArea() << " sq. units\n";

cout << "Circle B : Radius = " << B.getRadius() << " units, Area = "

<< B.getArea() << " sq. units\n";

cout << "Circle C : Radius = " << C.getRadius() << " units, Area = "

<< C.getArea() << " sq. units\n";

cout << "A == B : ";

if (A == B)

cout << "Circles are equal.\n";

else

cout << "Circles are not equal.\n";

cout << "A == C : ";

if (A == C)

cout << "Circles are equal.\n";

else

cout << "Circles are not equal.\n";

}

**QUESTION 9:** Write a program to overload new and delete operator in C++.

ANS:

#include <iostream>

#include <string>

class MyClass

{

float\* p= new float[100];

};

class MyClass2{

int five= 5;

std::string s= "hello";

};

int main()

{

int\* myInt= new int(1998);

double\* myDouble= new double(3.14);

double\* myDoubleArray= new double[2]{1.1,1.2};

MyClass\* myClass= new MyClass;

MyClass2\* myClass2= new MyClass2;

delete myDouble;

delete [] myDoubleArray;

delete myClass;

delete myClass2;

// getInfo();

}

**QUESTION 10:** Write a program to overload == operator to check whether two strings are equal or not.

ANS:

include<iostream>

#include<stdio.h>

#include<string.h>

using namespace std;

class String

{

char str[20];

public:

void getdata()

{

gets(str);

}

int operator ==(String s)

{

if(!strcmp(str,s.str))

return 1;

return 0;

}

};

int main()

{

String s1,s2;

cout<<"Enter first string :: ";

s1.getdata();

cout<<"\nEnter second string :: ";

s2.getdata();

if(s1==s2)

{

cout<<"\nStrigs are Equal\n";

}

else

{

cout<<"\nStrings are Not Equal\n";

}

return 0;

}