OBJECT ORIENTED PROGRAMMING

LAB 5

• Name :HITU RAJ

• Roll no. :2005025

• Branch :CSE

/\*Q1 Create a class complex which stores real\_025 and imaginary part of a complex

number. Include all types of constructors and destructor. The destructor should

display a message about the destructor being invoked. Create objects using

different constructors and display them.\*/

#include <iostream>

using namespace std;

class complex

{

    int real\_025;

    int img\_025;

public:

    complex() //DEFAULT CONSTRUCTOR

    {

        real\_025 = 10;

        img\_025 = 20;

    }

    complex(int a, int b) //parameterized constr

    {

        real\_025 = a;

        img\_025 = b;

    }

    complex(const complex &c) //copy constructor

    {

        real\_025 = c.real\_025;

        img\_025 = c.img\_025;

    }

    void display()

    {

        cout << real\_025 << "+i" << img\_025;

    }

    ~complex()

    {

      cout<<"\n memory released";

    }

};

int main()

{

    int n;

    cout << "\npress 1 if you want to use default constructor"

            "\npress 2 if you want to use parameterzed constructor"

            "\npress 3 if you want to use copy constructor";

    cin >> n;

    switch (n)

    {

    case 1:

    {

        complex c1;

        c1.display();

        break;

    }

    case 2:

    {

        complex c1(2, 3);

        c1.display();

        break;

    }

    case 3:

    {

        complex c2;

        complex c1(c2);

        c1.display();

        break;

    }

    default:

        break;

    }

    return 0;

}

**OUTPUT\_Q1**



/\*Q2 Create a class which stores time1 in hh:mm format. Include all the constructors.

The parameterized constructor should initialize the minute value to zero, if it

is not provided.\*/

#include <iostream>

using namespace std;

class time1

{

    int hr\_025;

    int min\_025;

public:

    time1() //default cons

    {

        hr\_025 = 10;

        min\_025 = 20;

    }

    time1(int t, int j = 0) //parameterized

    {

        hr\_025 = t;

        min\_025 = j;

    }

    time1(const time1 &a) //copy cons

    {

        hr\_025 = a.hr\_025;

        min\_025 = a.min\_025;

    }

    void display()

    {

        cout << hr\_025 << ":" << min\_025;

    }

};

int main()

{

    int a, call = 0;

    ;

    cout << "\npress 1 to create obj using default constructor"

         << "\npress 2 to create obj using paramerized constructor"

         << "\npress 3 to create obj using copy constructor";

    cin >> a;

    switch (a)

    {

    case 1:

    {

        time1 b;

        b.display();

        break;

    }

    case 2:

    {

    x:

        int x, y;

        cout << "enter the value of hr\_025 and min\_025\n";

        cin >> x >> y;

        if (call == 0)

        {

            time1 c(x, y);

            c.display();

            time1 d(c);

        }

        else

        {

            cout << "the value copied\n";

            time1 c(x, y);

            time1 d(c);

            d.display();

        }

        break;

    }

    case 3:

    {

        call = 1;

        goto x;

    }

    default:

        break;

    }

    return 0;

}

**OUTPUT\_Q2**



/\*Q3Create a class which stores a string1 and its length\_025 as data members. Include all

the constructors. Include a member function to join two string1s and display the

concatenated string1.\*/

#include <iostream>

#include <string.h>

using namespace std;

class string1

{

    char str\_025[10];

    int length\_025;

public:

    string1() //default

    {

        strcpy(str\_025, "hitu");

        length\_025 = strlen(str\_025);

    }

    string1(char a[]) //paramet

    {

        strcpy(str\_025, a);

        length\_025 = strlen(str\_025);

    }

    string1(const string1 &c)

    {

        strcpy(str\_025, c.str\_025); //cant write equal to!!!!

        length\_025 = c.length\_025;

    }

    void concatinate(string1 a, string1 b)

    {

        strcpy(str\_025, strcat(a.str\_025, b.str\_025));

        length\_025 = strlen(a.str\_025) + strlen(b.str\_025);

        cout << "the concated string is " << str\_025;

    }

    void display()

    {

        cout<<"the string is "<<str\_025;

    }

};

int main()

{   string1 a;

    char stri[10];

    int ca;

    cout << "press 1 to use default const\n"

         << "press 2 to use paramet const\n"

         << "press 3 to use copy const\n";

    cin >> ca;

    switch (ca)

    {

    case 1:

    {

        break;

    }

    case 2:

    {

        cout << "enter a string";

        cin >> stri;

        string1 b(stri);

        break;

    }

    case 3:

    {

        string1 c(a);

        break;

    }

    default:

        break;

    }

   a.concatinate(a,a);

    return 0;

}

**OUTPUT\_Q3**



/\*Q4 Write a C++ program using class to dynamically allocate two integer arrays,

add them it to a third array and display all the arrays.

For(int I=0; I&lt;a.len;I++)

P[I]=a.p\_025[I];\*/

#include <iostream>

using namespace std;

class aray

{

    int \*p\_025;

    int length\_025;

public:

    void create(int a)

    {

        length\_025 = a;

        p\_025 = new int[a];

        cout << "enter data in array\n";

        for (int i = 0; i < a; i++)

        {

            cin >> p\_025[i];

        }

    }

    void display()

    {

        for (int i = 0; i < length\_025; i++)

        {

            cout << p\_025[i]<<" ";

        }

    }

    void add(aray a, aray b)

    {  length\_025=a.length\_025 + b.length\_025;

        p\_025 = new int[length\_025];

        int c = 0;

        for (int i = 0; i < a.length\_025; i++)

        {

            p\_025[i] = a.p\_025[i];

            c++;

        }

        for (int i = 0; i < b.length\_025; i++)

        {

            p\_025[c] = b.p\_025[i];

            c++;

        }

    }

};

int main()

{

    aray a, b, c;

    int n, m;

    cout << "how many elements you have in the 1st array\n";

    cin >> n;

    a.create(n);

    cout << "how many elements you have in the 2st array\n";

    cin >> m;

    b.create(m);

    cout<<"\n1st array is \n";

     a.display();

     cout<<"\n2nd array is \n";

     b.display();

    cout << "\nadded array is\n";

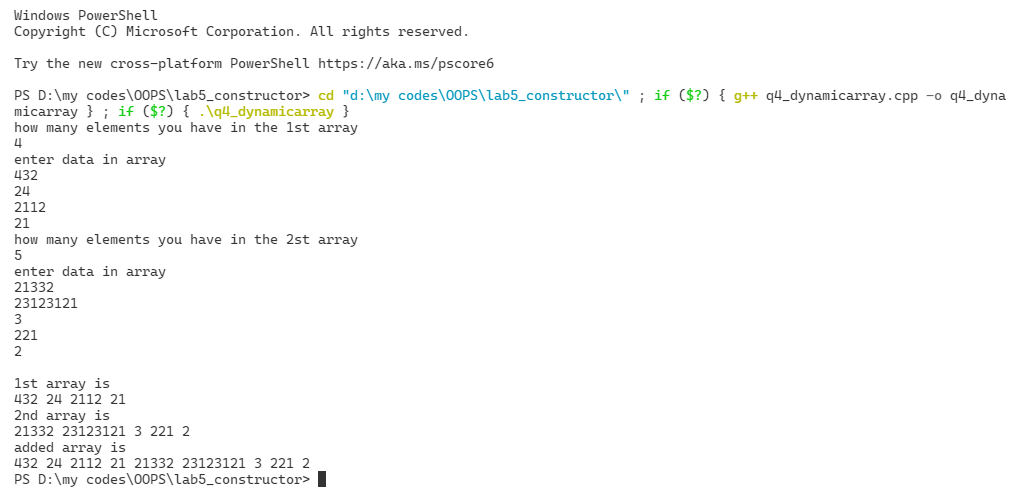
     c.add(a,b);

    c.display();

    return 0;

}

**OUTPUT\_Q4**



/\*Q5  WAP to demonstrate the order of call of constructors and destructors for a

class.\*/

#include <iostream>

using namespace std;

class test

{

    static int count\_025;

public:

    test()

    {

        cout << "constructor for obj" << count\_025 << "is called\n";

        count\_025++;

    }

    ~test()

    {

        cout << "destructor for obj" << --count\_025 << "is called\n";

    }

};

int test::count\_025=1;

int main()

{

    test a,b,c,d,e;

    return 0;

}

**OUTPUT\_Q5**



/\*Q6 WAP to count number of objects created from a class using concept of static

data members and static member function./

Class test{

Int I;

Static int count;

Public:

Test(){count++;}

Test(int k) { I=k; count ++;}

Static void print(){ cout&lt;&lt;” “&lt;&lt;count; }

};

int test:: count;

\*/

#include<iostream>

using namespace std;

class count

{   int i;

    static int Count\_025;

    public:

    count()

    {

        Count\_025++;

    }

    count(int k)

    {

        i=k;

        Count\_025++;

    }

    static void display()

    {

        cout<<"no. of times object created is "<<Count\_025<<" times";

    }

};

int count::Count\_025;

int main()

{

    count a,b,c;

    count f(23);

    count::display();

    return 0;

}

**OUTPUT\_Q6**



/\*Q7 .A book shop maintains the inventory of books that are being sold at the

workshop. The list includes details such as author\_025, title\_025, price\_025, publisher\_025 and stock\_025

position. Whenever a customer wants a book, the sales person inputs the title\_025 and

author\_025 and the system searches the list and displays whether it is available or not. If it

is not, an appropriate message is displayed. If it is, then the system displays the book

details and requests for the number of copies required. If the requested copies are

available, the total cost of their quested copies is displayed otherwise the message

“Required copies not in stock” is displayed. WAP using a class called Books with

suitable member functions and constructors.\*/

#include<iostream>

#include<string.h>

using namespace std;

class Books

{

    string author\_025, title\_025, publisher\_025;

    int price\_025, stock\_025;

public:

    Books()

    {

    }

    Books(string a, string t, string p, int m, int s)

    {

        author\_025 = a;

        title\_025 = t;

        publisher\_025 = p;

        price\_025 = m;

        stock\_025 = s;

    }

    friend void searchBooks(Books \*b, int n, string t, string a);

};

void searchBooks(Books \*b, int n, string t, string a)

{

    for (int i = 0; i < n; i++)

    {

        if (b[i].author\_025 == a && b[i].title\_025 == t)

        {

            if (b[i].stock\_025 <= 0)

            {

                cout << "The required book is not available!! Sorry!!\n";

                return;

            }

            else

            {

                cout << b[i].title\_025 << " by " << b[i].author\_025 << " published by " << b[i].publisher\_025 << "\n";

                cout << "Cost : " << b[i].price\_025 << "\n";

                cout << "Enter the required amount of books :- ";

                int no;

                cin >> no;

                if (b[i].stock\_025 >= no)

                {

                    int bill = no \* b[i].price\_025;

                    cout << "Your bill is :- " << bill << "\n";

                    b[i].stock\_025 -= no;

                    cout << "Visit again :)\n";

                    return;

                }

                else

                {

                    cout << "Sorry!! We don't have required amount of this book. Visit again :)\n";

                    return;

                }

            }

        }

        if (i == n - 1)

        {

            cout << "Never heard of this book. Sorry!!\n";

            cout << "Visit again :)\n";

            return;

        }

    }

}

int main()

{

    int n;

    cout << " How many books u have : ";

    cin >> n;

    Books \*b = new Books[n];

    for (int i = 0; i < n; i++)

    {

        cout << "\n======================================================== ========\n";

        string a, t, p;

        int m, s;

        cout << "\nEnter the price\_025 and stock\_025 of the book :- ";

        cin >> m >> s;

        getchar();

        cout << "\nEnter the name of the author\_025 of the book :- ";

        getline(cin, a);

      ///  cout << "\n";

        cout << "\nEnter the name of the book :- ";

        getline(cin, t);

        cout << "\nEnter the publisher\_025 name of the book :- ";

        getline(cin, p);

        b[i] = Books(a, t, p, m, s);

    }

    string st, sa;

    cout << "\n======================================================== ========\n";

    cout << "Enter the name of the book you wish to search :- ";

    getline(cin, st);

    cout << "Enter the name of the author\_025 of the book you wish to search :- ";

    getline(cin, sa);

    cout << "\n======================================================== ========\n";

    searchBooks(b, n, st, sa);

    cout << "\n======================================================== ========\n";

    cout << "The Program is terminated successfully!! ;)\n";

    return 0;

}

**OUTPUT\_Q7**

