1. Write a class Circle to implement

* Set value to radius
* Calculate area
* Calculate circumference
* float getRadius(); // returns radius
* float getArea();
* float getCircumference();

1. Write a class Circle to implement

* Default constructor
* One argument constructor
* Destructor
* Resize(Radius)
* Default value parameter constructor

1. Implement class Stack using the below code skeleton

constint STACK\_MAX 100;

class Stack {

private:

int data[STACK\_MAX];

int size;

public:

Stack() { // Constructor

// Complete the code

}

~Stack() { } // Destructor

intisStackFull(){}

intisStackEmpty(){}

int Top() {

// Complete the code

}

void Push(int d) {

// Complete the code

}

void Pop() {

// Complete the code

}

};

/\* \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Complete the following programs

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

1. Complete the below program

#include<iostream>

using namespace std;

class Vector

{

int \*v;

intsz;

public:

voidVectorsize(int size ) {

sz=size;

v=new int[size]; //Dynamic memory allocation for pointer v

}

void read(void);

voidshowsum(void);

void release(void) { // To free memory

delete []v;

}

};

main()

{

Vector obj;

int count;

cout<<"\nHow many elements are there in Vector?\n";

cin>>count;

obj.Vectorsize(count);

obj.read();

obj.showsum();

obj.release(); //Explicitly freeing memory

}

1. Complete the below program

#include <cstring>

#include<iostream>

using namespace std;

class Person

{

private:

int age;

char \*name;

public:

int getAge () const

{ return age;}

const char \*getName() const

{ return name; }

Person (char \* name = NULL, int age =0)

{

this->name=new char[strlen(name)+1];

strcpy(this->name,name);

this->age=age;

}

Person & operator= (const Person & other);

Person (const Person &other);

void show()

{

cout<<name<<"::"<<age<<endl;

}

};

Person &Person::operator= (const Person &other)

{

//write code here

}

Person::Person (const Person & other)

{

\*this=other; //OK, use user-defined assignment operator is invoked

}

void main()

{

Person name1("Prashanth",55);

name1.show();

Person name2("Praveen",66);

name2.show();

name2=name2;

name2.show();

cout<<"testing"<<endl;

}

‘

1. Complete the below program

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* File name: \*

\* Version No: \*

\* Description : program to illustrate Passing objects by value \*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include<iostream>

using namespace std;

class Distance

{

private:

float feet;

float inches;

public:

void read(void) {

}

void show(void) {

}

void add(const Distance &D1,const Distance &D2)

{

}

};

main()

{

Distance d1,d2,d3;

//Read elements to object d1

d1.read();

d1.show();

//Read elements to object d2

d2.read();

d2.show();

d3.add(d1,d2); //passing objects by value

cout<<"\nd3=d1+d2:";

d3.show();

}

1. Complete the below program

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* File name: student.cpp

\* Version No \*

\* Description : program to illustrate array of objects

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

# include <iostream>

using namespace std;

class student{

char name[25];

int RollNo;

int Marks;

public :

void ReadData(void);

voidPrintData(void);

intSearchData(student & r) const;

void update(student& r) ;

};

void student :: ReadData(void){

//Read values from stdin

}

void student :: PrintData(void)

{

}

int student :: SearchData(student & r) const{

if(!(strcmp(name,r.name)) &&RollNo == r.RollNo)

return 1;

else

return 0;

}

void student :: update(student& r) {

}

main()

{

student \*obj;;

obj = new student[10];

int i;

for(i = 0; i< 2 ;i++)

(\*(obj+i)).ReadData();

cout<<"Enter old Record to be updated\n";

student temp;

temp.ReadData();

for(i = 0; i< 2 ;i++)

if(obj[i].SearchData(temp)){

cout<<"Enter new Record to be updated\n";

student rec;

rec.ReadData();

obj[i].update(rec);

obj[i].PrintData();

break;

}

else

cout<<"Record Not found\n";

}

1. Complete the below program

main()

{

student obj[10];

int i;

for(i = 0; i< 2 ;i++)

obj[i].ReadData();

cout<<"Enter old Record to be updated\n";

student temp;

temp.ReadData();

for(i = 0; i< 2 ;i++)

if(obj[i].SearchData(temp)){

cout<<"Enter new Record to be updated\n";

student rec;

rec.ReadData();

obj[i].update(rec);

obj[i].PrintData();

break;

}

else

cout<<"Record Not found\n";

}

1. Complete the below program

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* File name: program7.cpp \*

\* Version No:1.3 \*

\* Description : Dynamic initialization through Copy constructors \*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include<iostream>

using namespace std;

class Vector

{

int \*v;

intsz;

public:

Vector(void){ v =NULL; sz=0;}

Vector (int size) {

}

Vector (Vector & v2); //Copy constructor

~Vector () {

delete []v;

}

int&elem (inti) {

if (i>= sz)

cout<< "Error:out of range\n";

return v[i];

}

void show ();

};

Vector::Vector (Vector &tempv) {

}

void Vector::show () {

}

main ()

{

inti;

Vector obj;

Vector v1 (5), v2 (5);

for (i = 0; i< 5; i++)

v1.elem(i) = i + 1;

// v1 = v2; // Bit wise copy is done

Vector v3 =v1; //Copy constructor is called v3(v2)

//Vector v3 = v2; //Copy constructor is called v3(v2)

cout<< "\nVector v1:";

v1.show ();

//cout<< "\nVector v2:";

//v2.show ();

cout<< "\nVector v3:";

v3.show ();

Vector v4 = v3; //Copy constructor is invoked v4(v3)

cout<<endl;

v4.show ();

}

1. Implement class Queue of dynamic size.

Constructors

Destructor

fnPush()

fnPop()

fnTop()

fnIsQueueFull()

fnIsQueueEmpty()

1. Circular link list with following behaviors

* Default Constructors, copy constructors, destructors
* Push\_front, push\_back, pop\_front, pop\_back functions
* Insert element at kth position, delete element from kth position.
* Search element in the list
* Find number of elements in the list

1. Doubly link list with following behaviors

* Default Constructors, copy constructors, destructors
* Push\_front, push\_back, pop\_front, pop\_back functions
* Insert element at kth position, delete element from kth position.
* Search element in the list
* Find number of elements in the list

1. Doubly circular link list with following behaviors

* Default Constructors, copy constructors, destructors
* Push\_front, push\_back, pop\_front, pop\_back functions
* Insert element at kth position, delete element from kth position.
* Search element in the list
* Find number of elements in the list

**Note: Q – 1-8 & 11-12🡪 are to be used for evaluation for 10 Marks [1 mark each]**