Assignment - 26 A Job Ready Bootcamp in C++, DSA and IOT MySirG

Member function, static, constructor

1. Define a class Complex to represent a complex number with instance variables a and

b to store real and imaginary parts. Also define following member functions

a. void setData(int,int)

b. void showData()

c. Complex add(Complex)

#include <iostream>

using namespace std;

class Complex

{

private:

int real, img;

public:

void setData(int x, int y)

{

real = x;

img = y;

}

void showData()

{

cout << "Real = " << real << " img = " << img << endl;

}

Complex add(Complex C)

{

Complex temp;

temp.real = real + C.real;

temp.img = img + C.img;

return temp;

}

};

int main()

{

Complex c1, c2, c3;

c1.setData(1, 2);

c2.setData(3, 4);

c1.showData();

c2.showData();

c3 = c1.add(c2);

c3.showData();

return 0;

}

2. Define a class Time to represent a time with instance variables h,m and s to store

hour, minute and second. Also define following member functions

a. void setTime(int,int,int)

b. void showTime()

c. void normalize()

d. Time add(Time)

#include <iostream>

using namespace std;

class Time

{

private:

int hour, minute, second;

public:

void setTime(int x, int y, int z)

{

hour = x;

minute = y;

second = z;

}

void showTime()

{

cout << "Hour = " << hour << " Minute = " << minute << " Second = " << second << endl;

}

void normalize()

{

}

Time add(Time t)

{

/\*

Time temp;

temp.hour = hour + t.hour;

temp.minute = minute + t.minute;

temp.second = second + t.second;

return temp;

\*/

Time temp;

int totalTime = hour \* 60 \* 60 + minute \* 60 + second + t.hour \* 60 \* 60 + t.minute \* 60 + t.second;

temp.hour = totalTime / (60 \* 60);

totalTime = totalTime % (60 \* 60);

temp.minute = totalTime / 60;

totalTime = totalTime % 60;

temp.second = totalTime;

return temp;

}

};

int main()

{

Time t1, t2, t3;

t1.setTime(3, 34, 23);

t2.setTime(6, 32, 5);

t1.showTime();

t2.showTime();

t3 = t1.add(t2);

t3.showTime();

return 0;

}

3. Define a class Cube and calculate Volume of Cube and initialise it using constructor.

#include <iostream>

using namespace std;

class Cube

{

private:

int a;

public:

Cube(int x)

{

a = x;

}

int showCubeVolume()

{

return a \* a \* a;

}

};

int main()

{

Cube c1(5);

cout << "Volmue of cube is " << c1.showCubeVolume();

return 0;

}

4. Define a class Counter and Write a program to Show Counter using Constructo

5. Define a class Date and write a program to Display Dateand initialise date object

using Constructors.

#include <iostream>

using namespace std;

class Date

{

private:

int dd, mm, yy;

public:

Date(int x, int y, int z)

{

dd = x;

mm = y;

yy = z;

}

void displayDate()

{

cout << "DD = " << dd << ", MM = " << mm << ", YY = " << yy << endl;

}

};

int main()

{

Date d(10,9,2022);

d.displayDate();

return 0;

}

6. Define a class student and write a program to enter student details using constructor

and define member function to display all the details.

#include <iostream>

#include <string.h>

using namespace std;

class Student

{

private:

int rollNo, age;

long int mobileNo;

char name[20];

public:

Student(int x, char studentName[], int y, int z)

{

rollNo = x;

strcpy(name, studentName);

age = y;

mobileNo = z;

}

void displayStudentData()

{

cout << "Roll No. - " << rollNo << ", Name - " << name << ", Age - " << age << ", Mobile No. - " << mobileNo << endl;

}

};

int main()

{

Student st(12, "Ajay", 19, 1234567890);

st.displayStudentData();

return 0;

}

7. Define a class Box and write a program to enter length, breadth and height and

initialise objects using constructor also define member functions to calculate volume

of the box.

#include <iostream>

using namespace std;

class Box

{

private:

int length, breath, height;

public:

Box(int x, int y, int z)

{

length = x;

breath = y;

height = z;

}

int volOfBox()

{

return length \* breath \* height;

}

};

int main()

{

Box b(1, 2, 3);

cout << "Volume of Box is: " << b.volOfBox() << endl;

return 0;

}

8. Define a class Bank and define member functions to read principal , rate of interest

and year. Another member functions to calculate simple interest and display it.

Initialise all details using constructor.

#include <iostream>

using namespace std;

class Bank

{

private:

int principal, roi, year;

public:

Bank(int x, int y, int z)

{

principal = x;

roi = y;

year = z;

}

int simpleIntrest()

{

return (principal \* roi \* year) / 100;

}

};

int main()

{

Bank b(1000, 2, 3);

cout << "simple intrest is: $" << b.simpleIntrest() << endl;

return 0;

}

9. Define a class Bill and define its member function get() to take detail of customer ,

calculateBill() function to calculate electricity bill using below tariff :

Upto 100 unit RS. 1.20 per unit

From 100 to 200 unit RS. 2 per unit

Above 200 units RS. 3 per unit.

#include <iostream>

using namespace std;

class Bill

{

private:

int cust\_id, consume\_unit;

char cust\_name[20];

public:

void get()

{

cout << "Enter customer id: ";

cin >> cust\_id;

cout << "Enter Customer Name: ";

fflush(stdin);

gets(cust\_name);

cout << "Enter total consume unit: ";

cin >> consume\_unit;

}

int calculateBill()

{

int totalRupess = 0;

if (consume\_unit <= 100)

{

totalRupess = consume\_unit \* 1.20f;

}

else if (consume\_unit <= 200)

{

totalRupess = (100 \* 1.20) + (consume\_unit - 100) \* 2.0f;

}

else if (consume\_unit > 200)

{

totalRupess = (100 \* 1.20) + (100 \* 2.0) + (consume\_unit - 200) \* 3.0f;

}

return totalRupess;

}

};

int main()

{

Bill b;

b.get();

cout << "Total Bill is: " << b.calculateBill();

return 0;

}

10. Define a class StaticCount and create a static variable. Increment this variable in a

function and call this 3 times and display the result.

#include <iostream>

using namespace std;

class StaticCount

{

private:

static int count;

public:

static int fun()

{

count++;

}

static int getCount()

{

return count;

}

};

int StaticCount::count = 0;

int main()

{

StaticCount::fun();

StaticCount::fun();

StaticCount::fun();

StaticCount::fun();

cout << "Calling times: " << StaticCount::getCount();

return 0;

}