**Question Number 1:**

**PROGRAM**

#include <iostream>

using namespace std;

class two;

class one

{

public:

void getter()

{

cout << "Enter the 1st Number : ";

cin >> num1;

}

void friend Friend\_Function(one, two);

private:

int num1;

};

class two

{

public:

void getter()

{

cout << "Enter the 2nd Number : ";

cin >> num2;

}

void friend Friend\_Function(one, two);

private:

int num2;

};

int main()

{

one obj1;

two obj2;

obj1.getter();

obj2.getter();

cout << endl;

Friend\_Function(obj1, obj2);

cout << endl << endl;

system("pause");

}

void Friend\_Function(one a, two b)

{

if (a.num1 > b.num2)

{

cout << a.num1 << " is the Greater Number !" << endl;

}

else if (a.num1 < b.num2)

{

cout << b.num2 << " is the Greater Number !" << endl;

}

else

cout << a.num1 << " is Equal to " << b.num2;

}

**A screenshot of a computer screen

Description automatically generated**

**Question Number 2:**

**PROGRAM**

#include <iostream>

using namespace std;

class B;

class A

{

public:

A()

{

x = 1;

}

friend B;

private:

int x;

};

class B

{

public:

void data(A obj1)

{

cout << "Enter any Number (From B to A) : ";

cin >> obj1.x;

cout << endl << endl << "Entered Number From B to A is : " << obj1.x;

}

};

int main()

{

A obj1;

B obj2;

obj2.data(obj1);

cout << endl << endl;

system("pause");

}

**A screenshot of a computer screen

Description automatically generated**

**Question Number 3:**

**PROGRAM**

#include <iostream>

#include<string>

using namespace std;

// Initiallisation of classes

class Mobile;

class Screen

{

public:

Screen()

{

length = 0, width = 0, height = 0;

}

friend Mobile;

private:

float length;

float width;

float height;

};

class Colour

{

public:

Colour()

{

mobile\_colour = "NULL";

}

friend Mobile;

private:

string mobile\_colour;

};

class Mobile

{

public:

void getter(Screen &OS, Colour &OC)

{

cout << "Enter the Length of Mobile : ";

cin >> OS.length;

cout << "Enter the Width of Mobile : ";

cin >> OS.width;

cout << "Enter the Height of Mobile : ";

cin >> OS.height;

cout << "Enter the Colour of Mobile : ";

cin >> OC.mobile\_colour;

}

void setter(Screen &OS, Colour &OC)

{

cout << "The Length of Mobile is : " << OS.length << endl;

cout << "The Width of Mobile is : " << OS.width << endl;

cout << "The Height of Mobile is : " << OS.height << endl;

cout << "The Colour of Mobile is : " << OC.mobile\_colour << endl;

}

};

int main()

{

int opt;

cout << "Enter the Number of Mobiles you want to consider : ";

cin >> opt;

Mobile \*M = new Mobile[opt];

Screen \*S = new Screen[opt];

Colour \*C = new Colour[opt];

cout << endl;

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

{

cout << "Enter the Values for !! " << i + 1 << " !! Mobile ! "<< endl;

M[i].getter(S[i] , C[i]);

cout << endl;

}

cout << endl;

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

{

cout << "The Values for !! " << i + 1 << " !! Mobile is : " << endl;

M[i].setter(S[i], C[i]);

cout << endl;

}

delete []M;

M = NULL;

delete []S;

S = NULL;

delete []C;

C = NULL;

cout << endl << endl;

system("pause");

}

**A screenshot of a computer screen

Description automatically generated**

**Question Number 4:**

**PROGRAM**

#include <iostream>

#include<string>

using namespace std;

// Initiallisation of classes

class Mobile;

class Screen

{

public:

Screen()

{

length = 0, width = 0, height = 0;

}

friend Mobile;

private:

float length;

float width;

float height;

};

class Colour

{

public:

Colour()

{

mobile\_colour = "NULL";

}

friend Mobile;

private:

string mobile\_colour;

};

class Mobile

{

public:

static int count;

void getter(Screen &OS, Colour &OC)

{

cout << "Enter the Length of Mobile : ";

cin >> OS.length;

cout << "Enter the Width of Mobile : ";

cin >> OS.width;

cout << "Enter the Height of Mobile : ";

cin >> OS.height;

cout << "Enter the Colour of Mobile : ";

cin >> OC.mobile\_colour;

++count;

}

void setter(Screen &OS, Colour &OC)

{

cout << "The Length of Mobile is : " << OS.length << endl;

cout << "The Width of Mobile is : " << OS.width << endl;

cout << "The Height of Mobile is : " << OS.height << endl;

cout << "The Colour of Mobile is : " << OC.mobile\_colour << endl;

}

};

int Mobile::count = 0;

int main()

{

int opt;

cout << "Enter the Number of Mobiles you want to consider : ";

cin >> opt;

Mobile \*M = new Mobile[opt];

Screen \*S = new Screen[opt];

Colour \*C = new Colour[opt];

cout << endl;

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

{

cout << "Enter the Values for !! " << i + 1 << " !! Mobile ! " << endl;

M[i].getter(S[i], C[i]);

cout << endl;

}

cout << endl;

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

{

cout << "The Values for !! " << i + 1 << " !! Mobile is : " << endl;

M[i].setter(S[i], C[i]);

cout << endl;

}

cout << "Total Number of iterations for the Mobile Function is : " << M[opt].count;

delete[]M;

M = NULL;

delete[]S;

S = NULL;

delete[]C;

C = NULL;

cout << endl << endl;

system("pause");

}

**A screenshot of a computer screen

Description automatically generated**

**Question Number 5:**

**PROGRAM**

#include<iostream>

using namespace std;

class DB;

class DM

{

public:

DM()

{

meters = 0, centi\_meters = 0;

}

void getter()

{

cout << "Enter the value of Distance in Meters : ";

cin >> meters;

cout << "Enter the value of Distance in Centi-Meters : ";

cin >> centi\_meters;

}

void setter()

{

cout << endl << "Distance Enter in Meters is : " << meters << endl;

cout << "Distance Enter in Centi-Meters is : " << centi\_meters << endl;

}

friend void Distance\_Addition(DM , DB);

private:

float meters;

float centi\_meters;

};

class DB

{

public:

DB()

{

feet = 0, inches = 0;

}

void getter()

{

cout << "Enter the value of Distance in Feet : ";

cin >> feet;

cout << "Enter the value of Distance in Inches : ";

cin >> inches;

}

void setter()

{

cout << endl << "Distance Enter in Feet is : " << feet << endl;

cout << "Distance Enter in Inches is : " << inches << endl;

}

friend void Distance\_Addition(DM , DB);

private:

float feet;

float inches;

};

int main()

{

DM M;

DB B;

cout << "Enter Distance in Following Terms !!" << endl;

M.getter();

B.getter();

cout << endl << "The Distance you Entered in Different Terms !!" << endl;

M.setter();

B.setter();

cout << endl << "Now Performing Calculations on Expressions !!" << endl;

Distance\_Addition(M,B);

cout << endl << endl;

system("pause");

}

void Distance\_Addition(DM M, DB B)

{

float total = 0;

total = M.meters + M.centi\_meters + B.feet + B.inches;

cout << endl << "Total Distance Becomes ! " << total << " ! after addition " << endl;

// For Calculation, I convert total distance into METERS ( total = Meters)

total = (B.feet / 3.3) + (M.centi\_meters / 10) + ((B.inches / 0.4) / 10) + M.meters;

cout << endl;

cout << endl << "Total Distance in Meters Becomes = " << total;

cout << endl << "Total Distance in Centi-Meters Becomes = " << total \* 10;

cout << endl << "Total Distance in Feet Becomes = " << total \* 3.3;

cout << endl << "Total Distance in inches Becomes = " << (total \* 10) \* 0.4;

}

**A screenshot of a computer screen

Description automatically generated**

**Question Number 6:**

**PROGRAM**

#include<iostream>

using namespace std;

class Teacher;

class Students

{

public:

void getter()

{

cout << "Enter Name of the Student : ";

cin >> name;

cout << "Enter ID of the Student : ";

cin >> id;

cout << "Enter GPA of the Student : ";

cin >> gpa;

}

void setter()

{

cout << endl << "Name of the Student is : " << name << endl;

cout << "ID of the Student is : " << id << endl;

cout << "GPA of the Student is : " << gpa << endl;

}

friend Teacher;

private:

char name[20];

char id[15];

float gpa;

};

class Teacher

{

public:

void GPA(Students \*S)

{

float choice = 0;

cout << "Enter New Gpa for " << S->name << " " << S->id << " whose current GPA is " << S->gpa << " : ";

cin >> choice;

S->gpa = choice;

}

};

int main()

{

Students \*S = new Students[10];

Teacher T;

// inputting values

for (int i = 0; i < 10; i++)

{

cout << "Enter the Data of " << i + 1 << " Student : " << endl;

S[i].getter();

cout << endl;

}

cout << endl; // chaining GPA of first 3 students

for (int i = 0; i < 3; i++)

{

T.GPA(&S[i]);

}

cout << endl; // outputting values

for (int i = 0; i < 10; i++)

{

cout << "The Data of " << i + 1 << " Student is : " << endl;

S[i].setter();

cout << endl;

}

cout << endl << endl;

system("pause");

}

A screenshot of a computer screen

Description automatically generated

A screenshot of a computer screen

Description automatically generated

**Question Number 7:**

**PROGRAM**

#include<iostream>

using namespace std;

class Vector;

class Point

{

public:

Point()

{

x = 0, y - 0, z = 0;

}

void getter()

{

cout << "Enter the Value of X-Axis for Point : ";

cin >> x;

cout << "Enter the Value of Y-Axis for Point : ";

cin >> y;

cout << "Enter the Value of Z-Axis for Point : ";

cin >> z;

}

void setter()

{

cout << endl << "The Value of X-Axix for Point is : " << x;

cout << endl << "The Value of Y-Axix for Point is : " << y;

cout << endl << "The Value of Z-Axix for Point is : " << z;

cout << endl << "Overall Point Becomes : (" << x << " , " << y << " , " << z << ")" << endl;

if (x >= 0 && y >= 0 && z >= 0)

{

cout << "Point's Direction is Positive along X , Y , Z axis " << endl;

}

else if (x< 0 && y >= 0 && z >= 0)

{

cout << "Point's Direction is Positive along Y , Z axis and Negative along X axis " << endl;

}

else if (x >= 0 && y < 0 && z >= 0)

{

cout << "Point's Direction is Positive along X , Z axis and Negative along Y axis " << endl;

}

else if (x >= 0 && y >= 0 && z < 0)

{

cout << "Point's Direction is Positive along X , Y axis and Negative along Z axis " << endl;

}

}

void friend New\_Position(Point, Vector);

private:

float x,y,z;

};

class Vector

{

public:

Vector()

{

X = 0, Y = 0, Z = 0;

}

void getter()

{

cout << "Enter the Value of X-Axis for Vector : ";

cin >> X;

cout << "Enter the Value of Y-Axis for Vector : ";

cin >> Y;

cout << "Enter the Value of Z-Axis for Vector : ";

cin >> Z;

}

void setter()

{

cout << endl << "The Value of X-Axix for Vector is : " << X;

cout << endl << "The Value of Y-Axix for Vector is : " << Y;

cout << endl << "The Value of Z-Axix for Vector is : " << Z;

cout << endl << "Overall Vector Becomes : (" << X << " , " << Y << " , " << Z << ")" << endl;

if (X >= 0 && Y >= 0 && Z >= 0)

{

cout << "Vector's Direction is Positive along X , Y , Z axis " << endl;

}

else if (X< 0 && Y >= 0 && Z >= 0)

{

cout << "Vector's Direction is Positive along Y , Z axis and Negative along X axis " << endl;

}

else if (X>= 0 && Y < 0 && Z >= 0)

{

cout << "Vector's Direction is Positive along X , Z axis and Negative along Y axis " << endl;

}

else if (X>= 0 && Y >= 0 && Z < 0)

{

cout << "Vector's Direction is Positive along X , Y axis and Negative along Z axis " << endl;

}

}

void friend New\_Position(Point, Vector);

private:

float X, Y, Z;

};

int main()

{

Point P;

Vector V;

cout << "Inputting Values for Point !!!" << endl;

P.getter();

cout << endl << "Inputting Values for Vector !!!" << endl;

V.getter();

cout << endl;

cout << "Outputting Values for Point !!!" << endl;

P.setter();

cout << endl << "Outputting Values for Vector !!!" << endl;

V.setter();

cout << endl << "After Adding Point and Vector, New Position Becomes !!!" << endl;

New\_Position(P, V);

cout << endl << endl;

system("pause");

}

void New\_Position(Point P, Vector V)

{

float a=0,b=0,c=0;

a = P.x + V.X;

b = P.y + V.Y;

c = P.z + V.Z;

cout << endl << "The Value of X-Axix for New Position is : " << a;

cout << endl << "The Value of Y-Axix for New Position is : " << b;

cout << endl << "The Value of Z-Axix for New Position is : " << c;

cout << endl << "Overall New Position Becomes : (" << a << " , " << b << " , " << c << ")" << endl;

if (a >= 0 && b >= 0 && c >= 0)

{

cout << "New Position's Direction is Positive along X , Y , Z axis " << endl;

}

else if (a < 0 && b >= 0 && c >= 0)

{

cout << "New Position's Direction is Positive along Y , Z axis and Negative along X axis " << endl;

}

else if (a >= 0 && b < 0 && c >= 0)

{

cout << "New Position's Direction is Positive along X , Z axis and Negative along Y axis " << endl;

}

else if (a >= 0 && b >= 0 && c < 0)

{

cout << "New Position's Direction is Positive along X , Y axis and Negative along Z axis " << endl;

}

}

**A screenshot of a computer monitor sitting on top of a computer

Description automatically generated**