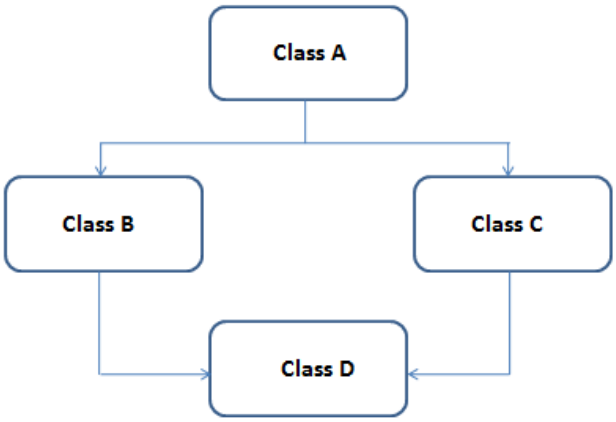
C++ Project



Class A:
Name: Robot
Variables: name, type weight, length, width, velocity, acceleration
Functions: getVelocity, getAcceleration

Class B:
Name: MobileRobot
Variables: numberOfWheels, list of sensors, cameraType, LidarType
Functions: getNumberOfWheels, getNumberOfSensors, move

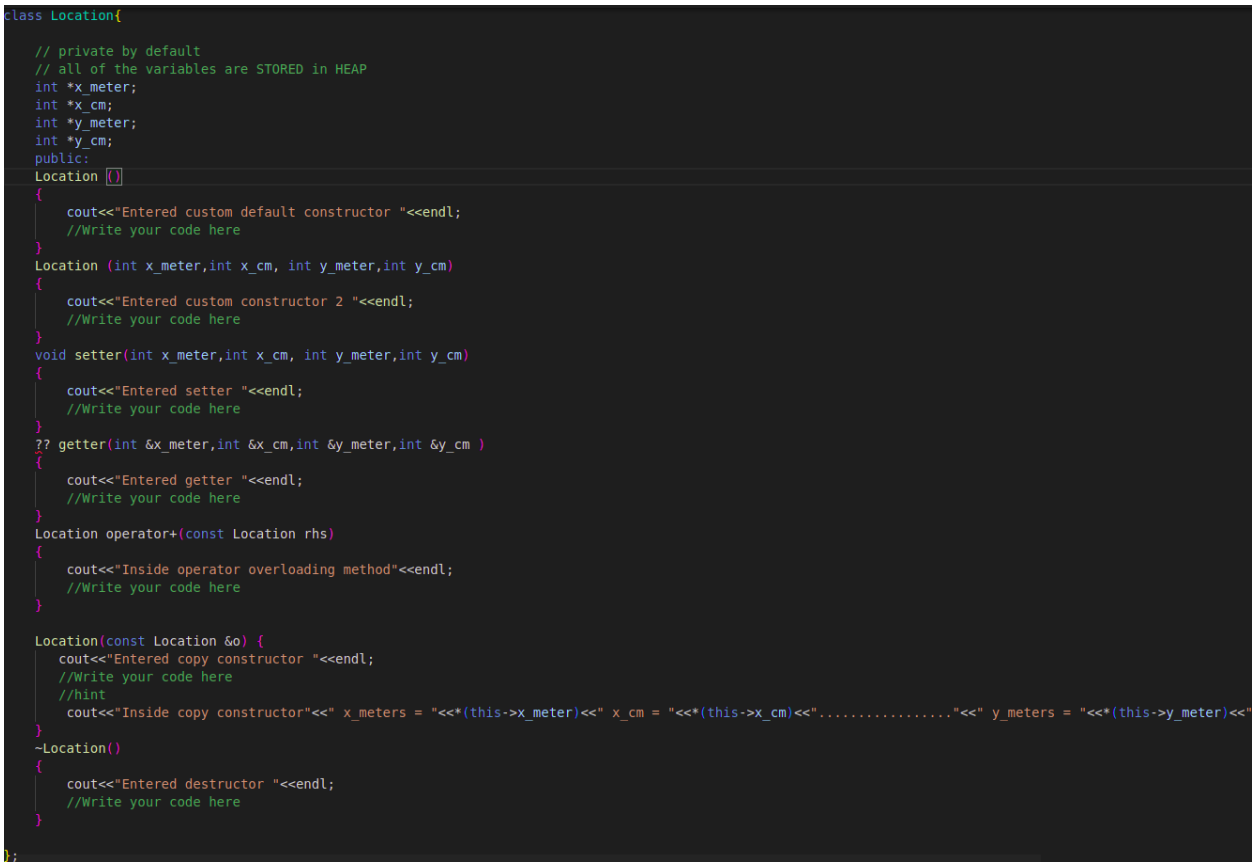
Class C:
Name: Manipulator
Variables: payload, workspace, maxReach, list of end effectors
Functions: getPayload, setEndEffectorType, getMaxReach, pickObject

Class D:
Name: MobileManipulator
Variables: range, weight, **Location(class E)**
Functions: navigate, move, pickObject, display (to display all the data)

Class E:
Name: Location
Variables: x_meter, x_cm, y_meter, y_m
Functions: getter, setter, operator+

Instructions:

- A is an abstract class.
- All classes must have 3 different types of constructors and one destructor (each has a message to display) as (“ Class name : message that describes the function ”) to clarify the running prosedural
- Demonstrate that all the constructors in MobileManipulator class are working and then call its functions (described below).
- E is a class that will be instantiated as an object attribute in class D, so each MobileManipulator has its own location that starts with (0,0,0,0) {x_meter,x_cm,y_meter,y_cm} and changes as the robot navigates; Ex: MobileManipulator.navigate(1,20,4,80).
- All attributes (variables) in class E “Location” are **POINTERS** and stored in **HEAP**.
- Kindly make sure that all of the methods (functions) in the “Location” class have the parameters as shown in the image below, **as well as the “cout” lines**.



```
class Location{

    // private by default
    // all of the variables are STORED in HEAP
    int *x_meter;
    int *x_cm;
    int *y_meter;
    int *y_cm;
    public:
    Location ()
    {
        cout<<"Entered custom default constructor "<<endl;
        //Write your code here
    }
    Location (int x_meter,int x_cm, int y_meter,int y_cm)
    {
        cout<<"Entered custom constructor 2 "<<endl;
        //Write your code here
    }
    void setter(int x_meter,int x_cm, int y_meter,int y_cm)
    {
        cout<<"Entered setter "<<endl;
        //Write your code here
    }
    ?? getter(int &x_meter,int &x_cm,int &y_meter,int &y_cm )
    {
        cout<<"Entered getter "<<endl;
        //Write your code here
    }
    Location operator+(const Location rhs)
    {
        cout<<"Inside operator overloading method"<<endl;
        //Write your code here
    }

    Location(const Location &o) {
       cout<<"Entered copy constructor "<<endl;
       //Write your code here
       //hint
        cout<<"Inside copy constructor"<<" x_meters = "<<*(this->x_meter)<<" x_cm = "<<*(this->x_cm)<<"................."<<" y_meters = "<<*(this->y_meter)<<"
    }
    ~Location()
    {
        cout<<"Entered destructor "<<endl;
        //Write your code here
    }

};
```

Results:

- Kindly run the code below in the main function (int main()):

```
// test MobileManipulator Constructors
1)MobileManipulator m1;
2)m1.display();
3)MobileManipulator m2("turtle", "waffle", 2, 2.5, 1.5, 2, 6);
4)m2.display();
5)MobileManipulator m3 (m2);

// test all the functions
6)m3.display();
7)m3.navigate(1,100,2,1);
// the results shown in the image below are for the line (1-7); you will
also need to run the following lines (8-9).
8)m3.pickObject();
9)m3.move();
```

- The output in the terminal should be as follow (w:

```
Entered custom default constructor
MobileManipulator : defult constructor
Mobile Manipulator data : name =  type =  weight = 1 velocity = 1 acceleration = 1 number of wheels = 2 payload = 1 range = 2
Entered custom default constructor
MobileManipulator : parameterized constructor
Mobile Manipulator data : name =  type =  weight = 1 velocity = 1 acceleration = 1 number of wheels = 2 payload = 6 range = 200
Entered custom default constructor
MobileManipulator : copy constructor
Mobile Manipulator data : name =  type =  weight = 1 velocity = 1 acceleration = 1 number of wheels = 2 payload = 6 range = 200
Entered getter
Location before modification x_meters = 0 x_cm = 0..... y_meters = 0 y_cm = 0
Entered custom constructor 2
Entered copy constructor
Inside copy constructor x_meters = 1 x_cm = 100..... y_meters = 2 y_cm = 1
Inside operator overloading method
Entered custom constructor 2
Entered destructor
Entered getter
Entered setter
navigated
Location modified x_meters = 2 x_cm = 0..... y_meters = 2 y_cm = 1
Entered destructor
Entered destructor
MobileManipulator : parameterized destructor
Entered destructor
MobileManipulator : parameterized destructor
Entered destructor
MobileManipulator : parameterized destructor
Entered destructor
```