[Ravi Patel] Instructor: Dr. Hindo

[CPSC 230]

Chapter 10- Homework (20 points)

Note: Submit your homework document in the inbox (chapter 10 HW).

Your name should be at the top left with the course code as above.

**Part 1 (8 pts.)**

Multiple Choice

1. In a structure definition, the identifiers declared in the braces are called **C**
   1. classes
   2. structs
   3. **member names**
   4. variables
2. You specify an individual member of a struct by using **D**
   1. the assignment operator
   2. an ampersand
   3. an underscore
   4. **The dot operator**
3. To assign values to a structure variable, you use the **B**
   1. equals operator
   2. **assignment operator**
   3. extraction operator
   4. less than operator
4. What is wrong with the following structure definition? **C**

struct MyStruct

{

int size;

float weight;

}

* 1. Nothing
  2. Can not have mixed data types in a structure
  3. **missing semicolon**
  4. Braces are not needed.

1. Given the following strucure definitions, what is the correct way to print the person's birth year? **A**

struct DateType

{

int day;

int month;

int year;

}

struct PersonType

{ int age;

float weight;

DateType birthday;

}

PersonType person;

* 1. **cout << person.birthday.year;**
  2. cout << year;
  3. cout << birthday.year;
  4. cout << peson.year;

1. Which of the following is the correct function definition header for the getAge function which is a member of the Person class? **D**
   1. int getAge();
   2. int getAge()
   3. int Person:getAge()
   4. **int Person::getAge()**
2. Given the following class definition and the following member function header, which is the correct way to output the private data? **C**

class Person

{

public:

void outputPerson(ostream& out);

private:

int age;

float weight;

int id;

};

void Person::outputPerson(ostream& out)

{ //what goes here?

}

* 1. out << person.age << person.weight << person.id;
  2. out << person;
  3. **out << age << weight << id;**
  4. outputPerson(person);

1. A member function of a class should be made private **C**
   1. always
   2. only if it will never be used
   3. **if it will only be used by other members of the class**
   4. never, it is illegal to make a member function private.
2. A member function that allow the user of the class to change the value in a data member is known as **A**
   1. **a mutator function**
   2. a mutation
   3. a manipulator function
   4. an accessor function
3. A Member function that allows the user of the class to see the value in a data member is known as **D**
   1. a mutator function
   2. a mutation
   3. a manipulator function
   4. **an accessor function**
4. If you design a class with private data members, and do not provide mutators and accessors, then **B**
   1. The private data members can still be accessed from outside the class by using the & operator
   2. **The data can not be changed or viewed by anyone.**
   3. None of the above
   4. A and B
5. A class member function that automatically initializes the data members of a class is called **C**
   1. the init function
   2. an operator
   3. **a constructor**
   4. a cast

**Part 2 (8 points)**

**Write main program to test the following class functions: getPoint, and CalDist. The test program includes getting two points and calculating the distance between two points.**

|  |  |
| --- | --- |
|  | /\*C++ program to create a class to get two points and calculate the distance between the two point \*/  #include  <iostream>  using namespace std;    class Distance  {      private:          int x;          int y;      public:          void getPoint();          double calDist( Distance &point1, Distance &point2 );   }; |

/\*C++ program to create a class to get two points and calculate the distance between the two point \*/

//CPSC 230 RAVI PATEL HW 10

#include <iostream>

using namespace std;

class Distance

{

private:

int x;

int y;

public:

void getPoint();

double calDist();

};

void Distance::getPoint(){

cout << "Enter the number for coordinate x: ";

cin >> x;

cout << "Enter the number for coordinate y: ";

cin >> y;

}

double Distance::calDist(){

return (y - x);

}

int main(){

Distance obj;

obj.getPoint();

cout << "Distance: " << obj.calDist() << endl;

}

**SAMPLE OUTPUT:**

**Enter the number for coordinate x: 1**

**Enter the number for coordinate y: 5**

**Distance: 4**