

Part I: Short Answer (4 questions, 18 points total)

Answer all of the following questions. READ EACH QUESTION CAREFULLY. Answer each question in the space provided on these pages. Budget your time so you spend enough on the programming questions at the end.

Keep your answers short and to the point. Good luck.

1. (4 points) We've stressed that it is important to hide implementation details by declaring them as private members of classes. Give two specific reasons why this is a good thing to do. ("It's a good idea" is not specific enough.)

2. (4 points) The following code computes the sum of two vectors by first reading the length of the vectors, reading the first one, then adding the second one to it and printing the result. Does it work properly, assuming the input contains correctly formatted data? If there are problems with the program, circle them, and give a brief explanation of what's wrong.

```
#include <iostream>
using namespace std;
int main() {
    int length;
    cout << "enter length: ";
    cin >> length;
    double *vector = new double[length];
    for (int k = 0; k < length; k++) {
        cin >> vector[k];
    }
    for (int k = 0; k < length; k++) {
        cin >> x;
        vector[k] = vector[k] + x;
    }
    for (int k = 0; k < length; k++) {
        cout << vector[k] << " ";
    }
    cout << endl;
    delete vector;
    return 0;
}
```

3. (4 points) Here is a short program that reads three integer values and prints them.

```
#include <iostream>
using namespace std;
int main() {
    int a = 1;
    int b = 2;
    int c = 3;
    cin >> a >> b >> c;
    cout << a << endl << b << endl << c << endl;
    return 0;
}
```

Now, suppose we execute this program and enter the following:

17 B52 143 10

What values will the program print?

4. (6 points) Here is a short program that does some pointer manipulation.

```
#include <iostream>
using namespace std;
int main() {
    int *p;
    int *q;
    int *r;

    p = new int;
    q = new int;
    r = p;
    *r = 10;
    *q = *p + 3;
    p = q;
    cout << *p << endl << *q << endl << *r << endl;

    

// code for part (b) goes here



    return 0;
}
```

(a) (3 points) Below, write the output that is produced when the “cout << ...” statement is executed. (If there is a problem that prevents execution from succeeding, describe what’s wrong.)

(b) (3 points) Add statements to the box in the code to delete all of the dynamically allocated storage that exists at that point in the program.

Part II. Programming Problem (2 questions, 26 points total)

5. (12 points) (This question has nothing to do with classes – only basic C++ data structures.)

Classroom Support Services keeps track of a lot of information about UW classrooms. They've hired you to help out with a project to keep track of information about classroom supplies. They are using the following data structure to represent information about the whiteboard supplies in a UW classroom.

```
struct RoomEntry {           // information about one classroom
    string  building;         // building abbreviation ("MGH", "HUB", etc.)
    int     roomNum;         // room number
    int     nMarkers;        // number of whiteboard markers in room
    int     nErasers;        // number of erasers in room
};

const int MAX_ROOMS = 472;   // maximum possible number of classrooms

struct Classrooms {          // description of all known classrooms
    int     size;            // current number of rooms in the list
    RoomEntry rooms[MAX_ROOMS]; // rooms[0..size-1] contains the room descriptions
};
```

Complete the definition of function `nErasersIn`, below, so it calculates and returns the total number of erasers in all classrooms that are in the specified building (i.e., find all the entries for classrooms in that building and add up the number of erasers). You should assume that appropriate libraries have already been `#included` (i.e., don't worry about `#includes`).

```
// = total number of erasers recorded in list r for all classrooms in the specified building
int nErasersIn(Classrooms r, string building) {
```

```
}
```

6. (14 points) This question involves classes `PlaceList` and `Location` from homework 2.

Give the implementation of a new member function in class `PlaceList` named `closestPlaceTo(...)`. This function should return the name of the place that is closest to (has the shortest distance from) the `Location` supplied as an argument. The specification of this new member function, which has been added to the definition of class `PlaceList` is:

```
// Return the name of the place in this PlaceList
// that is nearest to Location where
string closestPlaceTo(Location where);
```

Your function should return an empty string (“”) if the `PlaceList` is empty (has no entries).

Write your implementation of function `closestPlaceTo` below, as it would appear in the `PlaceList.cpp` implementation file. You only need to give the code for this function; don't worry about `#includes` or other details. Copies of the declarations of classes `PlaceList` and `Location` from the HW2 sample solution, including a new declaration for `closestPlaceTo` are given on the next page for reference.

```

const int MAX_PLACES = 30;          // maximum # places in the list

class PlaceList {                    // a list of place and location pairs
public:
    // Construct an empty PlaceList
    PlaceList();

    // Add place with given name and coordinates to this PlaceList
    // Assumption: this PlaceList is not full
    void add(double x, double y, string name);

    // = "this PlaceList contains a place with the specified name
    bool contains(string name);

    // = Location of the place with the specified name; if the name is
    // not found, the result is a Location whose coordinates are not defined.
    Location locationOf(string name);

    // Return the name of the place in this PlaceList
    // that is nearest to Location where
    string closestPlaceTo(Location where);

private:
    struct Place {                    // list entry for one place:
        string name;                  // place name
        Location loc;                 // place location
    };

    Place list[MAX_PLACES];           // information about places is stored
    int size;                         // in list[0..size-1]
    ""
};

class Location {
public:
    // Construct a Location with coordinates <x,y>
    Location(double x, double y);

    // Default constructor - initialize this Location to <0,0>
    Location();

    // = Distance from this Location to Location other
    double distanceTo(Location other);

    // = Direction from this Location to Location other, as a string:
    // north, northeast, east, southeast, south, southwest, west, northwest
    string directionTo(Location other);

private:
    double x, y;                     // x and y coordinates of this Location
};

```