Computer Science 24: Homework 2

Andrew Lu (6088157). Partner: Abel Semma.

Problem 1

Implement point3d.cpp.

```
#include "point3d.h"
#include <iostream>
using namespace std;

Point3d::Point3d(double x, double y, double z) {
        this -> x = x;
        this -> y = y;
        this -> z = z;
}

double Point3d::getX() const {
        return x;
}

double Point3d::getY() const {
        return y;
}

double Point3d::getZ() const {
        return z;
}

void Point3d::setX(double x) {
        this -> x = x;
}

void Point3d::setY(double y) {
        this -> y = y;
}

void Point3d::setZ(double z) {
        this -> z = z;
}

void Point3d::shift(int axis, double distance) {
        if(axis == 0) {
                x += distance;
        }
        else if(axis == 1) {
                y += distance;
        }
        else if(axis == 2) {
                z += distance;
        }
}

bool operator == (const Point3d& p1, const Point3d& p2) {
        //If the individual coordinates match, then the points are the same.
        if(p1.getX() == p2.getX() && p1.getY() == p2.getY() && p1.getZ() == p2.getZ()) {
                return true;
        }
        else {
                return false;
        }
}
```

Problem 2

Implement `bool operator == (bag& b1, bag& b2)`.

```
bool operator == (bag& b1, bag& b2) {
        //If the bag sizes are not the same, then the contents cannot match.
        if(b1.size() != b2.size()) {
                return false;
        }
        else {
                //Sort bags to be in numerical order.
                sort(b1);
                sort(b2);

                //If each element of the sorted bags match, then the bags are the same.
                //If, at any point, they do not match, then they are not the same.
                for(int i = 0, i < b1.size(), i++) {
                        if(b1.get_value(i) != b2.get_value(i)) {
                                return false;
                        }
                }

                return true;
        }
}
```

Problem 3

What is the output of the listed code samples?

Part A:

```
10 and 20
20 and 20
30 and 30
40 and 40
```

- The first two lines of code create new pointers that point to nothing.
- The next two lines create new integers and store their memory addresses in `p1` and `p2` respectively.
- The fifth and sixth lines dereference the two pointers and store the values 10 and 20, respectively.
- The seventh line dereferences the pointers `p1` and `p2` and prints their values, yielding the first output line: `10 and 20`.
- The eigth line deletes `p1`'s value, leaving `p1` as an empty pointer.
- The ninth line sets `p2`'s value in `p1`. Both `p1` and `p2` now point to the same int.
- The tenth line prints `p1` and `p2` again, yielding the second output line: `20 and 20`.
- The 11th and 13th lines dereference the pointers and store new values.
- The 12th and 14th lines print p1 and p2 again, yielding the third and fourth output lines: `30 and 30`, `40 and 40`.

Part B:

```
0 1 2 3 4 5 6 7 8 9
```

- The first line initializes an empty array of integers, `a`, with 10 elements.
- The second line stores the pointer to the first element of `a` in `p`.
- The third line initialzizes a counter variable `i`.
- The fourth line initializes a for loop that runs from values `i = 0` to `i = 9`, incrementing by one each iteration.
- The fifth line sets the `i` th element in `a` equal to `i` for each iteration of the loop.
- The sixth line initializes a for loop that runs from values `i = 0` to `i = 9`, incrementing by one each iteration.
- The seventh line dereferences `p` and prints the `i` th element followed by a space for each iteration of the loop.
- The eigth line adds an endl to the output.