CS250 Final
Yet More Review Questions

Spring 2013

Pointers

- Write a function isStrEqual that accepts two null terminated strings (type char *) and returns true if the two strings are identical; otherwise, false is returned. Use pointer notation.

1. What does a call to your function look like?
2. Write the function.

Pointers

- What is the difference between the two prototypes below?

```
void processArray (const char *pText);

void processArray (char * const pText);
```

- Which should be used for the previous function? Why?

Pointers

- What is **this**?
- Where does **this** come from?
- What is ***this**?

Word Interface

- Consider the interface for Word where mData is **NOT** null-terminated. The number of actual characters contained in mData is held in the variable mLength.

```
class Word
{
  public:
    static const int MAX_SIZE = 256;
    Word (); // Set Word to size zero
    Word (const char *); // The char * passed to Word IS null terminated
    Word (const Word &);
    friend ostream &operator<< (ostream &, const Word &);

  private:
    char mData[MAX_SIZE];
    int mLength;
};
```

- Implement everything in public

Questions

- Will the following program compile? If so, what is the output? If not, why not?

```
#include "Word.h"
#include <iostream>

using namespace std;

int main ()
{
  Word cWord1 ("CS250"), cWord2 ("Review"), cWord3 = cWord1, cWord4;
  cWord4 = cWord2;

  cout << "Words" << endl;
  cout << "Word 1 = " << cWord1 << endl;
  cout << "Word 2 = " << cWord2 << endl;
  cout << "Word 3 = " << cWord3 << endl;
  cout << "Word 4 = " << cWord4 << endl;

  return EXIT_SUCCESS;
}
```

Questions

1. There is a difference in how C++ assigns a value for cWord3 and cWord4 below. Explain the difference in detail.

```
Word cWord1 ("CS250"), cWord2 ("Review"), cWord3 = cWord1, cWord4;
cWord4 = cWord2;
```

2. Is Word overloaded, overridden, or redefined. Explain.

Composition

1. Create a Dictionary class that is able to hold up to 1024 Words.

2. Implement a method add that accepts a Word and adds the word to the dictionary if the word isn’t already in the dictionary.

Point

- Consider the class Point

```
class Point
{
  public:
    Point (double = 0.0, double = 0.0);
    void setX (double);
    double getX () const;
    void setY (double);
    double getY () const;
    friend ostream &operator<< (ostream &, const Point &);

  private:
    double mX, mY;
};
```

Polygon

- A Polygon is an abstract class capable of holding up to 1024 points. Write the interface for Polygon that has appropriate constructor(s), an add method, an overloaded insertion operator, virtual functions draw and perimeter, and a pure virtual function area.

Triangle

- Triangle is a concrete class that inherits from Polygon and is to have all of the functionality of a Polygon as well an implementation for area. Also, a Triangle is to have a method isRight. Write the interface for Triangle.

- Important: Once a triangle is created, one shouldn’t be able to change the triangle into something else (say a Rectangle).

Rectangle

- Rectangle is a concrete class that inherits from Polygon and is to have all of the functionality of a Polygon as well an implementation for area. Write the interface for Rectangle.

- Important: Once a rectangle is created, one shouldn’t be able to change the rectangle into something else.

main

- Create an array of up to 25 Polygon pointers that can point to a Triangle or a Rectangle.

- Assuming that the array of polygon pointers is pointing to numPolygon objects, output the area for each object.