Practice Problems for PE04

For PE04, you'll have two different problems. For Problem 1 I'll give you a main function that makes five functions calls. You must write the prototypes for those functions. These are similar to the "input, output, input-output" parameter clicker questions from class. This problem will be worth 25 points (5 points per prototype.)

Problem 2, worth 25 points, will ask you to write a variety of functions using both input, input-output and output (reference) parameters. You'll also need to use selection and loops.

1 THE BINARY TO DECIMAL PROBLEM

Write a function named `binToDec()` that accepts an integer parameter whose digits are meant to represent binary (base-2) digits, and converts that integer's representation to decimal (base-10). The function should return the integer value before it was converted. For example, given this code:

```
int a = 101011;
int b = binToDec(a);
cout << "a->" << a << ", b->" << b << endl;
```

The output is: `a->43, b->101011`

Constraints: Do not use a string in your solution. Also do not use any built-in base conversion functions from the system libraries. You must program this by "hand".

2 THE BINARY TO DECIMAL PROBLEM

Write a function named `decToBin()` that accepts an integer parameter whose digits are meant to represent decimal (base-10) digits, and converts that integer to a representation of binary (base-2). The function should return the integer value before it was converted. For example, given this code:

```
int a = 43;
int b = decToBin(a);
cout << "a->" << a << ", b->" << b << endl;
```

The output is: `a->101011, b->43`

Constraints: Do not use a string in your solution. Also do not use any built-in base conversion functions from the system libraries. You must program this by "hand".

3 THE SWAP PAIRS PROBLEM

Write a function named `swapPairs()` that accepts a string reference as a parameter and modifies that string so that each pair of adjacent letters will be reversed. If the string has an odd number of letters, the last letter is unchanged. For example, if a string variable s stores `"example"`, the call of `swapPairs(s);` should change the string to `"xemalpe"`. If s had been `"hello there"`, the call would produce `"ehll ohtree"`.

The function should return the number of "swaps" made. (Remember that there are three assignments involved in each swap.) Do not use any string functions from the standard library other than `length()` and `at()` or `[]`.

4 THE CRAZY CAPS PROBLEM

Write a function named `crazyCaps()` that accepts a string as a output parameter and changes that string to have its capitalization altered such that the characters at even indexes are all in lowercase and odd indexes are all in uppercase. For example, if a variable `s` stores `"Hey!! THERE!"`, then the call of `crazyCaps(s);` should change `s` to store `"hEy!! tHeRe!"`.

Do not use any string functions from the standard library other than `length()` and `at()` or `[]`. Remember that you can find the difference between upper and lowercase characters by subtracting `'A'` from `'a'`. Do not use the `topper()` or `tolower()` macros from `cctype`.

5 THE NAME DIAMOND PROBLEM

Write a function named `nameDiamond()` that accepts a string as an input parameter and returns a new string in a "diamond" format as shown below. For example, the call of

```
string s = nameDiamond("MARTY");
```

should return a string with embedded newlines, that looks like this when printed. Use `\n` for the embedded newlines. The last line should end with a newline.

```
M
MA
MAR
MART
MARTY
 ARTY
  RTY
   TY
    Y
```

6 THE FACTOR COUNT PROBLEM

Write a function named `factorCount()` that accepts an integer (assumed to be positive) as its input parameter, returns a count of its positive factors in its output parameter, and returns the largest factor (not counting `1` or the number itself), via the return statement. For example, the eight factors of `24` are `1`, `2`, `3`, `4`, `6`, `8`, `12`, and `24`, so the call `factorCount(24, fCount)` should return `12` and set `fCount` to `8`. If there are no factors other than `1` and the number itself (such as `3`), return `-1`.

7 The Leet String Problem

Write a function `strToLeet()` that accepts an input-output string parameter and converts the string to (or from) "leet speak" (aka 1337 speak), an internet dialect where various letters are replaced by other letters/numbers. The second, `bool` parameter, with a default value of `true`, will determine whether to translate to leet or to English. Return the converted string.

Original character	'Leet' character
o	0
l (lowercase L)	1
e	3
a	4
t	7
s (at the end of a word only)	Z

Here are some examples:

```
string s = "four score and";
cout << strToLeet(s) << endl;    // f0ur sc0r3 4nd
cout << str << endl;             // f0ur sc0r3 4nd
cout << strToLeet(s, false);     // four score and
cout << str << endl;             // four score and
```

8 The Body Mass Index Problem

Write a function named `bmiCalc()` that accepts two input parameters of type `double`, representing a person's `height` and `weight` (in that order). The function returns (through the `return` statement) the user's BMI. It also returns, through an `integer` output parameter, the BMI category as calculated below.

BMI	Category Class
below 18.5	1
18.5 – 24.9	2
25.0 – 29.9	3
30.0 and up	4

Here's an example, along with the expected output.

```
int bmiClass;
double bmi = bmiCalc(70.0, 194.25, bmiClass);
cout << "BMI = " << bmi << ", class " << bmiClass << endl;
// BMI = 27.8689, class 3
```

9 THE QUADRATIC PROBLEM

Write a function named `quadratic()` that computes roots of quadratic equations. Recall that a quadratic equation is one of the form, $ax^2 + bx + c = 0$.

Your function accepts five parameters:

- The integer coefficients `a`, `b`, and `c` as two input parameters
- Two real number (`double`) output parameters `root1` and `root2`.

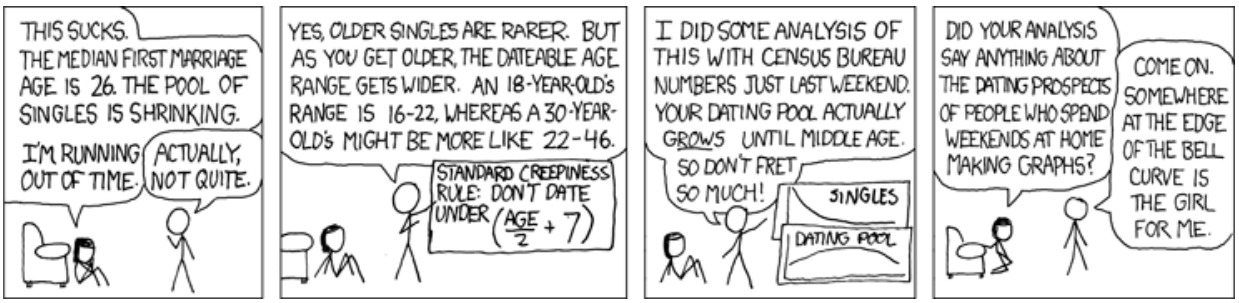
Your function should compute the two integer roots of the quadratic equation and store them into the two reference parameters. For example, the equation $x^2 - 3x - 4 = 0$ has roots of $x = 4$ and $x = -1$, so the call `quadratic(1, -3, -4, root1, root2);` should set `root1` to 4 and `root2` to -1. You may assume that the function has two real roots.

Recall the quadratic formula is:

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

10 THE DATING RANGE PROBLEM

Write a function named `datingRange()` that accepts three parameters: an integer input parameter for a person's age, and two integer output parameters for a minimum and maximum. The function should fills the `min`/`max` integers with the person's xkcd "dating range" as described in the following web comic strip:



Your minimum xkcd dating age is half your own age plus 7. Your maximum xkcd dating range is your own age, minus 7, then doubled. For example, the call `datingRange(48, min, max);` sets min to 31 and max to 82. You may assume that the age value passed is a non-negative integer.

11 The Making Change Problem

Write a function named `makeChange()` that takes two double input parameters, `cost` and `amount`, along with three `int` output parameters: `quarters` (25 cents), `dimes` (10 cents) and `cents`. We'll skip using nickels (5 cent coins) for this problem.

The `cost` is the cost of your purchases, and the `amount` is the amount you give to the cashier. Your function will calculate the change after subtracting the cost of the purchases from the amount. The output variables `quarters`, `dimes` and `cents` will be used for to return the coins and the `return` statement will be used to return the dollars.

Here's an example. Your purchases are $1.08 and you pay with a $5.00 bill. Your change is:

```
int quarters, dimes, cents;
int dollars = makeChange(1.08, 5.00, quarters, dimes, cents);
cout << dollars << " dollars, " << quarters << " quarters, "
    << dimes << " dimes, and " << cents << " cents" << endl;
// 3 dollars, 3 quarters, 1 dimes, and 7 cents
```

12 The Digit Range Problem

Write a function named `digitRange()` that accepts an integer as an input parameter and returns the range of values of its digits. The range is defined as 1 more than the difference between the largest and smallest digit value. For example, the call `digitRange(68437)` would return `6` because the largest digit value is `8` and the smallest is `3`, so `8 - 3 + 1 = 6`. If the number contains only one digit, return `1`. Solve this problem without using a string.

13 The Min-Max Digit Problem

Write a function named `minMaxDigit()` that accepts an integer as an input parameter and returns the largest and smallest digits using the two output parameters min and max. For example, the call `minMaxDigit(68437, min, max)` would set min to 3 and max to 8. If there is only one digit, then both min and max are set to the same value. The function has no return statement.

14 The Greatest Common Denominator Problem

Write a function named `gcd()` that accepts two integers as input parameters and returns the greatest common divisor of the two numbers. The greatest common divisor (`GCD`) of two integers, `a` and `b`, is the largest integer that is a factor of both `a` and `b`. The GCD of any number and `1` is `1`, and the `GCD` of any number and `0` is that number.

One efficient way to compute the `GCD` of two numbers is to use Euclid's algorithm, which states the following:

```
GCD(A, B) = GCD(B, A % B)
GCD(A, 0) = Absolute value of A
```

In other words, if you repeatedly take the remainder of `A` divided by `B` and then swap the two values, eventually `B` will store `0` and `A` will store the greatest common divisor.

For example: `gcd(24, 84)` returns `12`, `gcd(105, 45)` returns `15`, and `gcd(0, 8)` returns `8`.

15 The Digit Swap Problem

Write a function named `swapDigitPairs()` that accepts a positive integer `n` as an input-output parameter which is changed to a new value similar to n's but with each pair of digits swapped in order. For example:

```
int n = 482596;
int old = swapDigitPairs(n);
cout << "n->" << n << ", old->" << old << endl;
```

This returns `482597` but changes `n` to `845269`. Notice that the `9` and `6` are swapped, as are the `2` and `5`, and the `4` and `8`. If the number contains an odd number of digits, leave the leftmost digit in its original place. For example:

```
n = 1234567;
old = swapDigitPairs(n);
cout << "n->" << n << ", old->" << old << endl;
```

This converts `n` into `1325476`. Solve this problem without using a string.

16 The Min-Max String Problem

Write a function named `minMaxStr()` that accepts a string input parameter and two integer output parameters, `min` and `max`. Set `min` the ASCII value of the smallest alphabetical character and `max` to the largest. Only consider alphabetical characters. The `return` statement should return the real ratio of `min` to `max`. Make sure that the input string is not modified.