The Heritage Computer Challenge *2010*

C++ Division

Heritage High School
Newport News, Virginia

Welcome

Welcome to the Heritage Computer Challenge for 2010! You are to be commended for taking the time and making the effort to be here today. Have a great time and may all your programming efforts be successful!

~Mr. Charles F. Monroe, Contest Director

Instructions

The problems for this contest appear on the following pages, listed in order of difficulty. The maximum number of points you can earn is indicated under the title to each problem.

Problems are designed in the format used by The Great Computer Challenge, held annually each Spring at Old Dominion University. Some of these problems were actually used at the Great Computer Challenge in previous years.

Remember, this is a timed contest. Therefore, it is unlikely that you or anyone else will have time to complete all 5 problems in the allotted time. The winners will be the persons who earn the most points. You must earn at least 1 point to place.

Each solution should be saved as a project on your personal drive K **in a folder whose name is IDENTICAL to the project file name** (minus the extension). Source file names are up to you, as long as their extension is **.cpp**.

List of problems

Monroe Numbers	10 points
Three Sailors and a Monkey	10 points
Cryptarithm	20 points
Pyramid of Letters (Take Three)	20 points
Perfect Numbers	30 points

Monroe Numbers
(10 points)

Save in folder named: **MonroeNumbers**

You have heard of Fibonacci numbers, but have you ever heard of Monroe numbers? The first Monroe number is 0. The second one is 0. And the third Monroe number is 1. From then on, each Monroe number is the sum of the previous 3 Monroe numbers. Write a C++ program to allow the user to indicate which Monroe number he/she would like to view (1 to 39 please) and then display the requested Monroe number. Thus, the 29th Monroe number would be 4700770.

To get full credit, your program must find the 39th Monroe number as quickly as it finds the 29th.

Three Sailors and a Monkey
(10 points)

Save in folder named: **Sailors**

Three sailors, shipwrecked with a monkey on a desert island, have gathered on one day a pile of coconuts that are to be divided early the next day. Sometime during the night, one sailor arises, divides the pile into three equal parts, and finds one coconut left over, which he gives to the monkey. He then hides his share, and returns the remaining coconuts to a single pile. Later during the same night, each of the other two sailors arises separately and repeats the performance of the first sailor. In the morning all three sailors arise, divide the pile into three equal shares, and find one coconut left over, which they give to the monkey.

Write a program in C++ that will compute how many coconuts were in the original pile. Since there is more than one correct solution, the program should consider all coconut piles in the range of 1 to 1000. The output should be displayed in the console window and consist of the following:

a. The number of coconuts in the original pile.
b. The number of coconuts after each sailor removes a third.

Display your solutions in the following format:

	Orig-inal Pile	After 1st Sailor	After 2nd Sailor	After 3rd Sailor
1	79	52	34	22
2	160	106	70	46

Only the first two solutions are shown, so you can check your work.

Cryptarithm
(20 points)

Save in folder named: **Crypt**

In a cryptarithm puzzle letters have been substituted for digits in an arithmetic calculation. Each distinct letter is represented by a particular, but unique digit. Code a program to find all the solutions to the following cryptarithm.

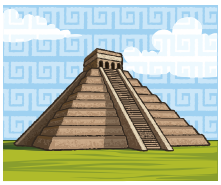
```
    OIL
     NO
    OIL
+    ON
_______
   LINO
```

Note that the numbers of the solution(s) can not have leading zeros. The output should display the above cryptarithm followed by the solution(s) in the same format.

Pyramid of Letters (Take Three)
(20 points)

Save in folder named: **Pyramid**

The Pyramid is a structure found in many cultures. This shape is often associated with supernatural power. You are asked to write a program that accepts a single character from "A" through "K" and produces an output in the shape of a pyramid composed of the letters up to and including the letter that was input. The top letter in the pyramid should be an "A". Each level below should add two more letters to the ending of the row and shift the row one position to the left so that the center column consists of the letters "A" through the letter that the user input. The number of letters in the last row should be one less than twice the number of rows.

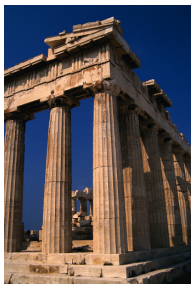


EXAMPLE:

```
Please enter the letter of choice: E

Your pyramid is as follows:

    A
   ABC
  ABCDE
 ABCDEFG
ABCDEFGHI

Are there more letters? Enter Yes or No
```

Perfect Numbers
(30 points)

Save in folder named: **PerfectNumbers**



The Greeks began an examination of numerology by classifying all positive integers as either perfect, abundant, or deficient. This classification scheme is based on the factors (even divisors) of the number. If the sum of all of the factors of a number (excluding the number itself) equals the number then it is said to be "perfect". For example, the factors of 6 are 1, 2, 3, and 6. Therefore, the number 6 is a perfect number. The total of the factors of 6 (excluding the number itself, in this case 6) is 1+2+3 = 6.

An abundant number is one in which this sum of factors (excluding the number itself) is greater than the number. An example of an abundant number is 12, because the sum of the factors of 12 is greater than 12. ex. 1+2+3+4+6 = 16 which is greater than 12. All numbers that are neither perfect nor abundant are deficient.

Write a program that prompts the user to enter a positive integer (allow integer values between 1 and 500). The program should at this point display the original number, the factors in that number and whether the number is perfect, abundant, or deficient.

EXAMPLE: (underlined values denote user input)

```
Please enter a positive integer:  6
The factors of 6 are: 1, 2, 3, and 6
The number 6 is perfect.

Please enter a positive integer: 12
The factors of 12 are: 1, 2, 3, 4, 6, and 12
The number 12 is abundant.

Please enter a positive integer: 333
The factors of 333 are: 1, 3, 9, 37, 111, and 333
The number 333 is deficient.
```