The Heritage Computer Challenge
2008
Heritage High School
Newport News, Virginia
C++ Division

Welcome

Welcome to the Heritage Computer Challenge for 2008! 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.

Solutions 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
Holler for a Dollar	20 points
Diamond Letters	20 points
Hex Clock	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.

Holler for a Dollar
(20 points)

Save in folder named: **Dollar**

Have the user enter two monetary amounts, each greater than 500.00 and then display their sum in correct monetary format with a leading dollar sign, commas inserted after thousands and millions, and always showing two digits for cents even if cents is a multiple of 10 or is 0. Input continues until either entered value is 500 or less.

Sample run:

```
Holler for a Dollar.  By Mr. C. Monroe.
Enter first monetary amount: 622.95
Enter second monetary amount: 824.47
The total is $1,447.42.

Enter first monetary amount: 622.40
Enter second monetary amount: 824.30
The total is $1,446.40.

Enter first monetary amount: 622.40
Enter second monetary amount: 824.60
The total is $1,445.00.

Enter first monetary amount: 1234567.08
Enter second monetary amount: 8765432.01
The total is $9,999,999.09.

Enter first monetary amount: 622.40
Enter second monetary amount: 5
Good-bye.

Press any key to continue...
```

Diamond Letters
(20 points)

Save in folder named: **Diamond**

Did you know astronomers have found a star that is made of pure diamond? Anybody have a good starship for sale cheap? Meanwhile, while we are waiting for somebody for perfect star travel, you are asked to write a program that accepts a single character from "A" through "Z" and produces an output in the shape of a diamond composed of the letters up to and including the letter that was input. The top letter in the diamond should be an "A" and on each level, the next letter in the alphabet should fall between the letter that was introduced in the level above it. After reaching and displaying the row containing the input letter, continue again in reverse with the other rows to complete the diamond.

Input continues until the user enters N.

EXAMPLE:

```
Please enter the letter of choice: E

Your diamond is as follows:

    A
   ABA
  ABCBA
 ABCDCBA
ABCDEDCBA
 ABCDCBA
  ABCBA
   ABA
    A

Display another diamond (Y or N)? _
```

Hex Clock
(30 points)

Save in folder named: **HexClock**

The clock is one of the oldest human inventions. There have been many kinds of clocks throughout history—water clocks, rope clocks, sundials, and more recently, analog (round) clocks and digital clocks. But all these clocks have one thing in common—they divide each day into 24 hours, each hour into 60 minutes, and each minute into 60 seconds. Depending on your country or your military status, your clock might display 24 hours or 12 hours twice using a.m. for the first half of the day and p.m. for the second half. Other than that, all of our clocks are basically the same.

In the 1850s, Swedish-American inventor John W. Nystrom proposed a hexadecimal system of weights and measures that included a hexadecimal clock. Sadly, his idea never caught on. Now that we use computers so much, many of our calculations would have been much easier.

The hex clock arranges time in a more consistent and logical manner than our present clock. Instead of dividing the day into arbitrary divisions (twenty-four hours in a day, sixty minutes in an hour, etc.), the hex clock represents time of day as a single hexadecimal number.

On a hex clock, each day is divided into 10_{16} (hex for 16) "hex hours" numbered 0 through F. Each "hex hour" is divided into 10_{16} (hex for 16) "hex minutes" also numbered 0 through F. And each "hex minute" is divided into 10_{16} (hex for 16) "hex seconds" also numbered 0 through F. When A_{16} (hex for 10) hex hours, D_{16} (hex for 13) hex minutes, and 9_{16} (hex for 9) seconds have passed, the time is written "A_D9" with an underscore placed between the hex hour and the hex minute.

Your job is to write a C++ program that will allow the user to input regular time in 24-hour format using semicolons. Seconds should be optional, so the user might enter "9:25" or "9:25:30". The program should then convert regular time to hex time and display it using the hex time format described above ("h_ms" where h is the hex hour, m is the hex minute, s is the hex second, and each is 0 through F).

If you cannot parse your input string to separate hours, minutes, and seconds out from between the semicolons, you may have the user input hours, minutes, and seconds separately, but you will lose 10 points for doing it that way.

Sample values so you can check your results:

regular time	hex time
0:00	0_00
12:00	8_00
18:00:00	C_00

regular time	hex time
9:30	6_55
4:35:15	3_0E
22:33:44	F_0A

regular time	hex time
14:15:16	9_80
23:42:09	F_CD
23:59:59	F_FF