

Western Carolina University

Department of Mathematics and Computer Science

Computer Science Programming Contest

May 4, 1999

Criteria for Determining Team Scores

Each program should have the following features:

1. execute without error and solve the corresponding problem,
2. execute properly using reasonable alternative input values,
3. incorporate commonly accepted structured style avoiding confusing and unnecessary code,
4. be documented with comments and easily understandable variable names, and
5. has a start of program comment that includes your school name, team number, and student names

The most important feature is that each program should execute properly. The total accumulated score for all four programs represents the team score. Ties in team scores will be eliminated by re-examining program robustness and style. All the programs have the same weight in the scoring.

Program Submission Procedure

When you have completed a C++ or Pascal problem, you should submit it for grading using the submit program. Below is an example use of the submit program. This interaction assumes there is a file named **example** in the current directory. That file will be copied to the grading directory. You may resubmit a solution for a problem; the resubmission file will create a new version and not over write the original submission.

tinuviel:~[!]submit

Please enter the name of the file: example

Which language is this file written in:

- 1 Basic
- 2 Pascal
- 3 C++

Enter 1,2, or 3: 1

What team number are you on? :2

Which problem is this file a solution for? : 3

Please verify the following information:

Language: 1

Team #: 2

Problem #: 3

Is this correct? {type y or n}: y

Submitted.

tinuviel:~[!]

1)

Lethal Dose

A government research lab has concluded that an artificial sweetener commonly used in diet soda pop will cause death in laboratory mice. A friend of yours is desperate to lose weight but cannot give up soda pop. Your friend wants to know how much diet soda pop it is possible to drink without dying as a result. Write a program to supply the answer (i.e. the number of grams of sweetener that would be a lethal dose for your friend and the corresponding number of cans of diet soda). The input to the program is the amount in grams of artificial sweetener needed to kill a mouse, the weight in grams of the mouse, and the weight in grams of the dieter. To ensure the safety of your friend be sure the program requests the weight at which the dieter will stop dieting, rather than the dieter's current weight. Assume that diet soda contains 1/10th of one percent artificial sweetener and that there are 350 grams of soda per can of soda. A typical run follows:

```
$ ./a.out
Enter the weight of the mouse in grams
15
Enter the lethal dose for the mouse in grams
100
Enter the desired weight of the dieter, in grams
45400
For these parameters:
mouse weight: 15 grams
lethal dose for the mouse: 100 grams
Dieter weight: 45400 grams
The lethal dose in grams of sweetener is: 302667
Lethal number of cans of pop: 864762
```

2)

Day of the Week

There is a simple algorithm for determining the day of the week (Monday, Tuesday, ..., Sunday) for an arbitrary date (specified as a Day, Month, and Year). Here is the algorithm.

Step 1) Input three integers: Month (a value from 1 to 12), Day (a value from 1 to 31), and Year (a value greater than 0).

Step 2) If Month is less than 3 then add 12 to Month and subtract 1 from Year.

Step 3) Compute the following expression

$$\text{Day} + 2 * \text{Month} + \text{Integer Part of } (0.6 (\text{Month} + 1)) + \text{Year} + \\ \text{Integer Part of } (\text{Year} / 4) - \text{Integer Part of } (\text{Year} / 100) + \\ \text{Integer Part of } (\text{Year} / 400)$$

Step 4) When the above expression is divided by 7, a remainder of 0 indicates "Monday," 1 indicates "Tuesday," 3 indicates "Wednesday" and so on.

Write a program that has a Month, Day, and Year as its input and outputs the name of the day of the week for that date. Two sample inputs and outputs follow.

Input: 1 2 7 1941 Note: this is December 7, 1941 which was Pearl Harbor Day
Output: Sunday

Input: 5 4 1999 Note: this is today, the day of the CS Contest
Output: Tuesday

3)

Greatest Common Denominator

The greatest common divisor of two positive integers x and y is the largest positive integer d which (evenly) divides both x and y . Write a program that prompts a user to input two positive integers x and y . The program should compute and output the greatest common divisor of x and y . The program should let the user repeat this process until he chooses to quit, and then terminate gracefully.

Example

Input: 12, 30

Output: 6 is the greatest common divisor of 12 and 30.

Input: 10, 40

Output: 10 is the greatest common divisor of 10 and 40.

Input: 15, 8

Output: 1 is the greatest common divisor of 15 and 8.

4)

Coin Flips

Suppose that you flipped a coin. The coin is equally likely to land heads or tails. If you flipped the coin `numFlips` times, what would be the longest run of consecutive heads and what would be the longest run of consecutive tails? For example, in the sequence `HTHTTTTHHHT` the longest run of consecutive heads is of length 4 and the longest run of consecutive tails is of length 3. Write a program to find out. The program's only input is the number of coin flips to perform. The program's output is the sequence of heads and tails that occurs and the size of the longest run of consecutive heads and the size of the longest run of consecutive tails. To complete this program you need to simulate a coin flip that is equally likely to be a head or a tail. C++, Pascal, and BASIC all support such a simulation through use of a library function.

- **C++:** In C++ the function is called `random()`. The function `random()` returns an integer between 0 and a large value, so you will have to use the mod operator to make the result either 0 or 1. In particular, the expression `random() % 2` returns randomly either a 0 or a 1 which you can consider to represent a head and a tail, respectively. You can use `random()` in your program as long as your program has the statement `#include <stdlib.h>`. Your program will always generate the same random sequence.

Example output for five coin flips is:

```
tinuviel:~/[1] ./a.out
Enter the number of coin flips: 5
Tail
Head
Tail
Tail
Tail
The longest run of heads is 1
The longest run of tails is 3
```

- **Pascal:** The Pascal function `GPC_RandInt(2)` randomly returns a 0 or a 1 which you can consider to represent a head and a tail, respectively. You can use `GPC_RandInt(2)` in your program as long as your program has the following statements at its start. Your program will always generate the same random sequence.

```
#include <gpc.pas> program
head(input,output);
```

```
uses gpc;
```

Example output for five coin flips is:

```
tinuviel:~/[1]/a.out
Enter the number of coin flips.
5
Tail
Head
Tail
Head
Head
The max number of heads in a run is 2
```

The max number of tails in a run is 1

- **BASIC:** Near the start of your program have the statement

```
RANDOMIZE
```

to start the random number generator. The QBASIC function call *int (2 * rnd)* will then randomly return a 0 or a 1 which you can consider to represent a head and a tail, respectively. Your program will always generate the same random sequence. At the question about the random-number seed, just hit the Enter key.

Example output for five coin flips is:

```
Random-number seed (-32768 to 32767)?
Enter the number of coin flips: 5
Heads
Tails
Tails
Heads
Tails
Longest run of heads is: 1
Longest run of tails is: 2
```