

CSci 1113 Spring 2007
Lab Midterm — 90 minutes
Section 12: Thursday, 5:25

In your home directory, create a new directory called `midterm` and change into that directory using the following commands:

```
cd
mkdir midterm
cd midterm
```

Do all your work for the lab midterm exam within this directory.

The lab midterm will consist of three problems. Save your source code for the three problems in files named `midtermA.cpp`, `midtermB.cpp`, and `midtermC.cpp`, respectively. Do not overwrite or delete your source code, even if you have compiled it and have an executable file.

A. (30 points) Running Speed

In this problem you will write a program named `midterm.cpp` to compute the speed of a runner in a 100 meter race.

Write a C++ program that allows the user to input a 100 meter time. The user should also input whether they want the speed reported in miles per hour ('m') or kilometers per hour ('k'). The program should then compute and print out the speed in the desired form.

For example, the women's 100 meter record is 10.49 seconds, which gives a speed of 34.32 kilometers per hour. Note the speed should be reported to the hundredths place.

A kilometer is 1000 meters; a mile is 1609 meters.

To compile Program A use

```
g++ midtermA.cpp -o A
```

This will produce an executable with a unique name so you will not have to recompile all three programs during grading. To execute the program simply type `A` at the command prompt.

B. (35 points) Speed Using a Function

In this part you will write a program that uses a function to compute the speed, and allows the user to find speeds for different 100 meter times in a single execution of the program.

Create a file called `midtermB.cpp`. This program will be somewhat similar to your program in Part A, but will have the following differences:

- First, turn your procedure for miles per hour speed into a function `find_speed`. This function should return the speed in miles per hour as a double. It should take as an argument the seconds input by the user for the 100 meter time. Your main program should first have the user input the 100 meter time information, then call the function to compute the speed in miles per hour, and then (in the main program) print out the speed.
- Second, the program should allow the user to repeat this process for as many 100 meter times as he or she wishes, quitting only if the user inputs a negative time for the 100 meter time.

Be sure to include a function *declaration* for the function.

To compile Program B use

```
g++ midtermB.cpp -o B
```

This will produce an executable with a unique name so you will not have to recompile all three programs during grading. To execute the program simply type `B` at the command prompt.

C. (35 points) File I/O Problem — Computer Company Growth

Create a file called `midtermC.cpp` for your source code.

The file `data12.dat` contains information about the (former) computer company Control Data. The first row in the file contains information for 1958, the second for 1960, the third for 1962, etc. Note the years are in 2 year increments. Each row contains 3 numbers. The first is the year. The second is the company revenue, in millions of US dollars. The third is the number of profit, again in millions of US dollars. The final line contains the three numbers `-1 -1 -1` to indicate the end of the file. Use the following command to copy the data file to your current directory:

```
cp ~barry/data12.dat data12.dat
```

Suppose you want to classify the line in this data file into three categories:

- Years where both the profits and revenue increased from the previous 2-year period.
- Years where both the profits and revenue decreased from the previous 2-year period.
- Other years — the profit decreased, but revenue increased from the previous 2-year period, or the profit increased, but revenue decreased from the previous 2-year period.

Then write a C++ program that opens the file and counts and prints out the number of lines for each category above.

```
Profits increase, revenues increase: xxx
Profits decrease, revenues decrease: xxx
Other years:                          xxx
```

where `xxx` should be filled in with the correct counts. Note the counts should be aligned into a column.

Be sure to check for a successful file connection and close the file when you're done using it. You may assume the file contains at least three lines of data, and that it does not contain any erroneous data. Do not assume you know in advance how many items are in the file — your program should check for the sentinel values `-1` at the end of the file.

Use the following command to compile Problem C. It will produce an executable with a unique name so you will not have to recompile all three programs during grading:

```
g++ midtermC.cpp -o C
```

HAND IN THE ENTIRE EXAM, NOT JUST THE GRADING SHEET.

CSci 1113, Spring 2007: Lab Midterm Grading Sheet

You:	ID:		Name:		Section:	12
------	-----	--	-------	--	----------	-----------

<i>Problem</i>	<i>Part</i>	<i>Points Possible</i>	<i>Score</i>	<i>TA Initials</i>
A	#includes and program skeleton	2		
A	Variable declarations	4		
A	Data prompt/input	6		
A	if-else statement	5		
A	Calculations	6		
A	Data output	5		
A	Compiles/executes	2		
B	#includes and program skeleton	2		
B	main function (general)	6		
B	Loop in main	6		
B	find_speed function declaration	4		
B	find_speed function header	4		
B	find_speed function body	6		
B	find_speed function call	5		
B	Compiles/executes	2		
C	#includes and program skeleton	2		
C	Open the data file	4		
C	Check for successful open	2		
C	Loop to read the data file	8		
C	Calculations	9		
C	Data output/format	6		
C	Close the data file	2		
C	Compiles/executes	2		
Total		100		