Imagine a **tollbooth** at a bridge. Cars passing by the booth are expected to **pay a hlala** toll. sometimes a car goes by without paying.

The tollbooth keeps track of the number of cars that have gone by, and of the total amount of money collected.

لنتخيل ان لدينا كشكات لتحصيل الرسوم على السيارات التي تمر على جسر

كل سياره تمر يجب عليها ان تدفع 50 هلله كرسوم. يحصل في بعض المرات ان تمر بعض السيارات بدون ان تدفع .

هذه الكشكات تقوم بحساب عدد السيارات التي مرت بالاضافه لكمية المال التي تم جمعها .

قومي بعمل برنامج يمثل وظيفة هذه الكشكات كالتالي :

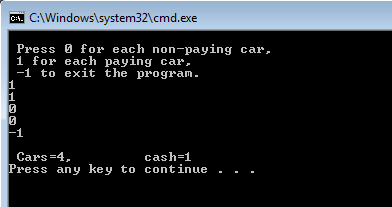
* Model this tollbooth with a **class** called **tollBooth**.
* The class contained **two** data items, the first one is from type int to hold the total number of cars.
* The second one is from type **double** to hold the total amount of money collected.
* A **constructor** initializes both of these variables to 0.
* A member function called **payingCar()** increments the car total and adds 0.50 to the cash total.
* A member function called **nopayCar**(), increments the car total but adds nothing to the cash total.
* A member function called **display()** displays the two totals.

Write a main() test this class.

This program should allow the user to enter (1) count a paying car, and( 0) to count a non-paying car.

Entering (-1) should cause the program to print out the total cars and total cash and then exit.

The run for the program could be like this:



**-3 for every girl come without the running program!**

Add the following:

* When the user enters (-1) the program calls the display function AND the print the results in a text file name “tollbooth.txt”.

Run the program multiple times to fill the file with bunch of data.

* Write another program to read from tollbooth file and print out its contents.

**Bonus:**

Write a full program, starting from #include, that gets the name of a text file from the user. The program then counts the number of words and lines in the file. You may assume the user's file exists and contains only words, no numbers or isolated punctuation marks. You may also assume that all line breaks occur immediately after a word and there are no blank lines. A sample run is shown below for the input file at right.

|  |  |
| --- | --- |
| Enter file name: input.txt  # words = 6  # lines = 3 | **input.txt** |
| Hello Sara.  How are you?  Good? |