Program 1: Trip Analysis...

Due date: Monday, October 16, 11:59pm.

1 Purpose

To write a program requiring use of variable declarations, reading values into and writing the values stored in variables, assignment statements and conditional statements.

Programming environment. This is a solo programming project. You are responsible for all the work related to the program development, testing and submission.

Collaboration. **Any** collaboration between peers, as well as any collaboration with outside sources is **strictly prohibited**. If you have any questions, concerning the assignment, please consult the instructor.

2 Program Description

You are going on a trip. On the way to your final destination, you pass by two checkpoints. At the beginning of the trip your velocity was zero. For each of the three legs of the trip (start to checkpoint 1, checkpoint 1 to checkpoint 2, checkpoint 2 to end) you will be given some information about the distance to the point, time it took you to travel there, your final speed (velocity) and/or your acceleration along the way.

Your program will compute the remaining parameters of your trip for each leg, and will output them. After the trip is over, the program will perform a cursory analysis of the trip and output some extra information.

A formal description of the program is given below.

Program Requirements.

TR0. Physics. Recall from high school that motion equations involve five parameters:

Parameter name	Notation
Starting velocity	u
Ending velocity	v
Acceleration	a
Displacement/distance travelled	d
Time in travel	t

Equations of motion, connect these five quantities in the following manner:

$$d = ut + \frac{at^2}{2}$$
$$v = u + at$$

Given any three quantities, we can compute the remaining two. The base equations above compute d and v from u, t and a. Inverting the first equation, we can get the following :

$$t = \frac{-u + \sqrt{u^2 + 2ad}}{a}$$ [1]

$$a = \frac{2(d - ut)}{t^2}$$

Invertng the second equation we can get:

$$a = \frac{v - u}{t}$$
$$t = \frac{v - u}{a}$$

(note that in all computations you need to perofrm for this project, you will always know the starting velocity u, but might not know some other quantities.)

TR1. Overview. Your program will consist of two parts: trip computer part and trip analysis part.

In the trip computer part, your program will read a number of inputs, defining some of the trip parameters for each consecutive leg of the trip and will compute and output the remaining parameters. It will also compute some overall trip information at the end.

In the trip analysis part your program will analyze various trip parameters and print the results of the analysis to screen.

TR2. Trip computer part overview. The trip computer part of the program tracks the trip as it proceeds through the three legs. Each leg ends at a checkpoint. There are five parameters that are used to describe each leg of the trip:

[1] We get this by solving the equation $\frac{at^2}{2} + ut - d = 0$ for t and using the positive root, as elapsed time is non-negative.

1. initial velocity
2. final velocity
3. acceleration
4. distance travelled
5. time in transit

At the beginning of each leg, you will know the value of one parameter, the initial velocity. It is set to 0 at the beginning of the first leg, and will be equal to the final velocity of the previous leg, on legs two and three of the trip. At each leg, your program will prompt the user for two other parameters (different pair for each leg). Upon obtaining these parameters, the program will compute the remaining two parameters using appropriate formulas described in requirement **TR0**. After that, the program outputs the computed values and proceeds to the next leg, or to the trip analysis part of the program (at the end of leg three).

The following table shows what information will be obtained from user input and what information will be computed.

Trip Leg	Known	Obtained from input	Computed
Leg 1: Start — Checkpoint #1	initial velocity	final velocity distance travelled	time in transit acceleration
Leg 2: Checkpoint #1 — Checkpoint #2	initial velocity	acceleration distance travelled	final velocity time in transit
Leg 3: Checkpoint #2 — Checkpoint #3	initial velocity	final velocity time in transit	acceleration distance travelled

We consider acceleration to be constant on each leg of the trip (this makes it possible for us to use the motion equations described in requirement **TR0**).

TR3. Program data. Your program shall have a number of declared variables to represent the velocities, times, distances and accelerations discussed below. All variables you declare in the proccgram must be of type `float`.

TR4. Units of measurement. Your program shall use the following units of measurement:

Parameter	Unit of measurement	Abbreviation
initial velocity, final velocity	*meters per second*	m/sec^2
acceleration	*meters per second per second*	m/sec^2
distance	*meters*	s
time	*seconds*	sec

T5. Output. All output of your program **must match exactly** the output of instructor's program given to you. This includes newlines, extra or missing spaces, and any extra or missing punctuation.

TR6. Leg 1. The initial velocity on Leg 1 of the trip is 0 m/s. Your program shall start by reporting the initial velocity:

```
Initial Velocity: 0 m/sec
```

All lines that are not input promts in the program shall end with newline character.

After the initial velocity is reported, the program shall print:

```
Enter distance to checkpoint #1:
```

The program shall read the distance to the checkpoint #1 from keyboard. After that, the program shall print:

```
Enter velocity at checkpoint #1:
```

The program shall read the final velocity at the end of Leg 1.

The program then shall compute the acceleration during Leg 2 and the time in transit, and shall output it to screen using the following text:

```
Checkpoint #1: acceleration = <acc> m/sec^2
Checkpoint #1: time elapsed = <time>  sec
```

TR7. Leg 2. The program shall print

```
Enter distance to checkpoint #2:
```

The program shall read the distance between checkpoints #1 and #2. It shall then print

```
Enter acceleration to checkpoint #2:
```

The program shall read the acceleration on Leg 2.

After that, the program shall compute the final velocity upon reaching checkpoint #2 and the time in transit. It the prints the acceleration during leg 2 and **the total (legs 1 and 2 combined)** time on travel. The output shall look as follows:

```
Checkpoint #2: time elapsed = <time> sec
Checkpoint #2: velocity = <velocity> m/sec
```

TR8. Leg 3. The program shall print

```
Enter time to checkpoint #3:
```

It shall read the time in transit from checkpoint #2 to checkpoint #3. Next, the program shall print

```
Enter velocity at checkpoint #3:
```

and read the final velocity at checkpoint #3.

The program shall then compute the distance travelled on Leg 3 of the trip and the acceleration on the leg. Both values shall be printed out using the text as follows:

```
Checkpoint #3: acceleration = <acc> m/sec^2
Checkpoint #3: distance to = <dist> m
```

This will complete the trip computer part of the program.

TR9. Trip analysis overview. Trip analysis part shall consist of two things. First, your program will compute and output the following three quantities:

- total distance travelled in all three legs;
- total time in transit;
- average velocity over all legs.

Next, your program will analyze the trip information to draw the following conclusions:

- How the speed of travel changed over time;
- Which leg of the trip was the longest;
- How grand was the entire trip;
- What were the means of transportation

It will output one line of text per each conclusion.

TR10. Trip totals computation. Your program shall compute the following three quantities:

1. Total distance of the trip. Add together the distances travelled on each leg.
2. Total time in transit. Add together the times in transit for each leg. Break the time into minutes and seconds. You will be reporting minutes and seconds separately in the output described below.
3. Average velocity. Total distance divided by total time.

Once the computation is complete, print two empty lines and output the computed values as follows:

```
Total distance travelled: <totalDistance> meters
Total time of travel: <totalMins> min <totalSec> sec
Average velocity: <avgVelocity> meters/sec
```

Print two more empty lines.

TR11. Trip Analysis. Part 1: speed of travel. Print to screen

```
Trip Analysis:
```

The first thing to analyze is the velocities at different periods of time during the trip. During the trip computer part of the program, you should obtain (from keyboard, or via computation) three velocities: the final velocities at each of the three checkpoints. We will refer to them as V1, V2 and V3 respectively.

Your program shall detect the following four situations, and output the appropriate text for each:

Situation	Output
V1 $>$ V2 $>$ V3	`You started fast, but then slowed down...`
V1 $<$ V2$<$V3	`You kept speeding up...`
V1 $=$ V2 $=$ V3	`Your speed was steady`
none of the above	`Your speed went up and down`

TR12. Trip Analysis. Part 2: Longest leg. Your program shall determine the longest leg of the trip. The length of the legs is measured by comparing the distances travelled on each leg. The following outputs should be provided for each possibility:

Situation	Output
Leg 1 is the longest:	`First leg of the trip was the longest`
Leg 2 is the longest:	`Second leg of the trip was the longest`
Leg 3 is the longest:	`Third leg of the trip was the longest`

TR13. Trip Analysis. Part 3: Length of travel. Your program shall determine analyze the overall length (total distance) of the travel and provide an appropriate message in each of the following situations:

Situation	Output
Trip was less than 100 meters long	`You travelled mere meters`
Trip was between 100 meters and 1km	`You travelled hundreds of meters`
Trip was 1km or longer	`You travelled kilometers upon kilometers!`

TR14. Trip Analysis. Part 4: Transportation mode. Your program shall look at the average velocity and make a guess about your mode of transportation as follows:

Situation	Output
Average velocity is less than 2 m/sec	`Looks like you walked`
Average velocity is between 2 and 6 m/sec	`Maybe you rode a horse...  Maybe you rode a bike...`
Average velocity is between 6 and 50 m/sec	`I think you drove a car`
none of the above	`You definetly flew in an airplane!`

This shall complete the work of your program.

TR15. Program name. Name your program `trip.c`.

TR16. Input guards. You are guaranteed that all entered velocities, distances and times will be non-negative (times will be positive, velocities and distances can be equal to 0).

Accelerations, both entered and computed can be negative.

During Leg 2, when asked to enter the acceleration, certain negative accelerations may lead to negative final velocities, i.e., negative accelerations may mean that checkpoint #2 is **cannot be reached**. Generally speaking, you are not responsible for this case. However, if you want to, you can check for this case (it affects only Leg 2 of the trip). If you discover that the velocity became negative as the result of negative acceleration, you may stop program execution and exit the program right away.

This case can be checked in two different ways. First, you can compute the final velocity, and check if it is non-negative. If it is not, you can stop the program. Second option is to check the *determinant* of the quadratic equation you need to solve in order to find time. If the determinant ($u^2 + 2ad$) is negative, i.e., $2ad < -u^2$, then the quadratic equation has no solutions. In this case you can stop the program.

Also, you may need to handle the case of acceleration being equal to 0 in a special way when writing Leg 2 code.

General Notes

Math. You are responsible for the remainder of the program design for this program. In particular, you are responsible for coming up with the correct math (physics) to compute the outputs of the program based on the inputs.

ANSI C. Your program shall be written in ANSI C. The instructor will compile your program using the following `gcc` flags:

```
gcc -ansi -Wall -Werror -lm
```

(note that you may need to use `sqrt` function from the `math.h` library package, hence `-lm` flag may be needed when you compile.)

Any program that does not compile in this fashion will be assigned a score of 0.

Style. Your code will checked for style. Your program shall conform to the style described at

http://users.csc.calpoly.edu/∼cstaley/General/CStyle.htm

In addition, the header comment shall be as described in Lab 2 specification.

Any style violations are subject to an automatic 10% penalty.

Testing. You are be provided with the instructor's binary, `trip-alex` and with a set of test cases for testing your program. We also include two standard testing scripts, `trip-tests.csh` for your program and `trip-tests-alex.csh` for the instructor's binary.

Please note, that you are expected to use both the instructor's binary, **and** the test cases and test scripts in your work.

Outputs of instructor's and your program **must match**. The only dispensation is given to rounding errors due to floating point computations.

Any program that fails any of the public tests will not receive more than 30% of the grade.

3 Submission Instructions

Submission.

Files to submit. You shall submit the `trip.c` file.

No other files can be submitted. In fact, if you submit *other* file, or submit one of the three files about with an *incorrect filename*, you will receive an email informing you about a submission error, and asking you to resubmit.

Submission procedure. You will be using `handin` program to submit your work. The procedure is as follows:

- `ssh` to `vogon` (`vogon.csc.calpoly.edu`). Submit using the following command:

  ```
  > handin dekhtyar program01 trip.c
  ```

Late submission. You may submit late for a 24-hour period following the deadline. Late submissions are subject to the standard 10%—30% penalty at the instructor's discretion.

4 Sample Output

```
$ ./trip
Initial Velocity: 0 m/sec
Enter distance to checkpoint #1:100
Enter velocity at checkpoint #1:5

Checkpoint #1: acceleration = 0.125000 m/sec^2
Checkpoint #1: time elapsed = 40.000000 sec

Enter distance to checkpoint #2: 200
Enter acceleration to checkpoint #2: 0.5

Checkpoint #2: time elapsed = 60.000000 sec
Checkpoint #2: velocity = 15.000000 m/sec

Enter time to checkpoint #3: 10
Enter velocity at checkpoint #3: 20

Checkpoint #3: acceleration = 0.500000 m/sec^2
Checkpoint #3: distance to = 175.000000 m


Total distance travelled: 475.000000 meters
Total time of travel: 1 min 10.000000 sec
Average velocity: 6.785714 meters/sec


Trip Analysis:
You kept speeding up...
Second leg of the trip was the longest
You travelled hundreds of meters
I think you drove a car

$ ./trip
Initial Velocity: 0 m/sec
Enter distance to checkpoint #1:1000
Enter velocity at checkpoint #1:10

Checkpoint #1: acceleration = 0.050000 m/sec^2
Checkpoint #1: time elapsed = 200.000000 sec

Enter distance to checkpoint #2: 500
Enter acceleration to checkpoint #2: 0

Checkpoint #2: time elapsed = 250.000000 sec
Checkpoint #2: velocity = 10.000000 m/sec

Enter time to checkpoint #3: 4
Enter velocity at checkpoint #3: 10

Checkpoint #3: acceleration = 0.000000 m/sec^2
```

```
Checkpoint #3: distance to = 40.000000 m


Total distance travelled: 1540.000000 meters
Total time of travel: 4 min 14.000000 sec
Average velocity: 6.062992 meters/sec


Trip Analysis:
Your speed was steady
First leg of the trip was the longest
You travelled kilometers upon kilometers!
I think you drove a car
```