Instructions Follow instructions *carefully*, failure to do so may result in points being deducted. Hand in all your source code files through webhandin and make sure your programs compile and run by using the webgrader interface. You can grade yourself and re-handin as many times as you wish up until the due date. Print a hardcopy of the rubric for this assignment and hand it in by the due date.

Partner Policy You may work in pairs for this assignment if you chose. If you do work in any groups or pairs, you must follow these guidelines:

1. You must work on *all* problems *together*. You may not simply partition the work between you.
2. You should not discuss problem details with other groups or individuals beyond general questions.
3. Hand in only one hard copy (and one soft copy) under the first author's name/cse login. Be sure to include both names.

Naming Instructions

- For problems 1, 3, and 5 implement the programs in C and place your source code in files named `investment.c`, `bearing.c`, and `halflife.c` respectively.
- For problems 2, 4, and 6 implement the programs in Java and place your source code in classes/files named `Odometer.java`, `TimeDilation`, and `CellPhone.java` respectively. Place all classes in the default package.

Programs

1. The ROI (Return On Investment) is computed by the following formula:

$$\text{ROI} = \frac{\text{Gain from Investment} - \text{Cost of Investment}}{\text{Cost of Investment}}$$

Write a program that prompts the user to enter the cost and gain (how much it was sold for) from an investment and computes and outputs the ROI. For example, if the user enters \$100,000 and \$120,000 respectively, the output look similar to the following.

```
Cost of Investment: $100000.00
Gain of Investment: $120000.00
Return on Investment: 20.00%
```

2. Write a program to compute the real cost of driving. Gas mileage is usually measured in miles per gallon,[1] but the real cost should be measured in how much it costs drive a mile, that is, dollars per mile. Write a program to assist a user in figuring out the real cost of driving. Prompt the user for the following inputs.

   - Beginning odometer reading
   - Ending odometer reading
   - Number of gallons it took to fill the tank
   - Cost of gas in dollars per gallon

   For example, if the user enters 50,125, 50,430, 10 (gallons), and $3.25 (per gallon), then your output should be something like the following.

   ```
   Miles driven: 305
   Miles per gallon: 30.50
   Cost per mile: $0.11
   ```

3. A *bearing* can be measured in degrees on the scale of $[0, 360)$ with $0°$ being due north, $90°$ due east, etc. The (initial) directional bearing from location A to location B can be computed using the following formula.

   $$\theta = \text{atan2}(\sin(\Delta) \cdot \cos(\varphi_2),\ \cos(\varphi_1) \cdot \sin(\varphi_2) - \sin(\varphi_1) \cdot \cos(\varphi_2) \cos(\Delta))$$

   Where

   - φ_1 is the latitude of location A
   - φ_2 is the latitude of location B
   - Δ is the difference between location B's longitude and location A's longitude
   - atan2 is the two-argument arctangent function

   Note: the formula above assumes that latitude and longitude are measured in radians r, $-\pi < r < \pi$. To convert from degrees d $(-180 < d < 180)$ to radians r, you can use the simple formula:

   $$r = \frac{d}{180}\pi$$

   Write a program to prompt a user for a latitude/longitude of two locations (an origin and a destination) and computes the directional bearing (in degrees) from the origin to the destination. For example, if the user enters: 40.8206, -96.7056 (40.8206° N, 96.7056° W) and 41.9483, -87.6556 (41.9483° N, 87.6556° W), your program should output something like the following.

   ```
   From (40.8206, -96.7056) to (41.9483, -87.6556):
   bearing 77.594671 degrees
   ```

[1] Only in the USA, though

4. General relativity tells us that time is relative to your velocity. As you approach the speed of light ($c = 299,792$ km/s), time slows down relative to objects traveling at a slower velocity. This *time dilation* is quantified by the Lorentz equation

$$t' = \frac{t}{\sqrt{1 - \frac{v^2}{c^2}}}$$

Where t is the time duration on the traveling space ship and t' is the time duration on the (say) Earth.

For example, if we were traveling at 50% the speed of light relative to Earth, one hour in our space ship ($t = 1$) would correspond to

$$t' = \frac{1}{\sqrt{1 - (.5)^2}} = 1.1547$$

hours on Earth (about 1 hour, 9.28 minutes).

Write a program that prompts the user for a a velocity which represents the *percentage* p of the speed of light (that is, $p = \frac{v}{c}$) and a time duration t in hours and outputs the relative time duration on Earth.

For example, if the user enters 0.5 and 1 respectively as in our example, it should output something *like* the following:

```
Traveling at 1 hour(s) in your space ship at
50.00% the speed of light, your friends on
Earth would experience:
1 hour(s)
9.28 minute(s)
```

Your output should be able to handle years, weeks, days, hours, and minutes. So if the user inputs something like 0.9999 and 168

```
Traveling at 168.00 hour(s) in your space ship at
99.99% the speed of light, your friends on
Earth would experience:
1 year(s)
18 week(s)
3 day(s)
17 hour(s)
41.46 minute(s)
```

5. Radioactive isotopes decay into other isotopes at a rate that is measured by a half-life, H. For example, Strontium-90 has a half-life of 28.79 years. If you started with 10 kilograms of Strontium-90, 28.79 years later you would have only 5 kilograms (with the remaining 5 kilograms being Yttrium-90 and Zirconium-90, Strontium-90's decay products).

Given a mass m of an isotope with half-life H we can determine how much of the isotope remains after y years using the formula,

$$r = m \cdot \left(\frac{1}{2}\right)^{(y/H)}$$

For example, if we have $m = 10$ kilograms of Strontium-90 with $H = 28.79$, after $y = 2$ years we would have

$$r = 10 \cdot \left(\frac{1}{2}\right)^{(2/28.79)} = 9.5298$$

kilograms of Strontium-90 left.

Write a program that prompts the user for an amount m (mass, in kilograms) of an isotope and its half-life H as well as a number of years y and outputs the amount of the isotope remaining after y years. For the example above your output should look something like the following.

```
Starting with 10.00kg of an isotope with half-life
28.79 years, after 2.00 years you would have
9.5298 kilograms left.
```

6. Write an app to help people track their cell phone usage. Cell phone plans for this particular company give you a certain number of minutes every 30 days which must be used or they are lost (no rollover). We want to track the average number of minutes used per day and inform the user if they are using too many minutes or can afford to use more.

   Write a program that prompts the user to enter the following pieces of data:

   - Number of minutes in the plan per 30 day period, m
   - The current day in the 30 day period, d
   - The total number of minutes used so far u

   The program should then compute whether the user is over, under, or right on the average daily usage under the plan. It should also inform them of how many minutes are left and how many, on average, they can use for the rest of the month. Of course, if they've run out of minutes, it should inform them of that too.

   For example, if the user enters $m = 250$, $d = 10$, and $u = 150$, your program should print out something similar to the following.

```
10 days used, 20 days remaining
Average daily use: 15 min/day

You are EXCEEDING your average daily use (8.33 min/day),
continuing this high usage, you'll exceed your minute plan by
200 minutes.

To stay below your minute plan, use no more than 5 min/day.
```

Of course, if the user is under their average daily use, a different message should be presented. You are allowed/encouraged to compute any other stats for the user that you feel would be useful.