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.

Naming Instructions Place your code in source files with the file names `refraction.c` and `rateOfReturn.c` respectively. For all of your programs, do some rudimentary input validation and exit the program on any erroneous input.

Programs

1. When light travels from one medium to another it may get slowed down. If the light is traveling at an angle, it may get *refracted*. That is, the angle at which the light enters the medium may get bent to a more extreme angle. This is depicted in Figure 1.

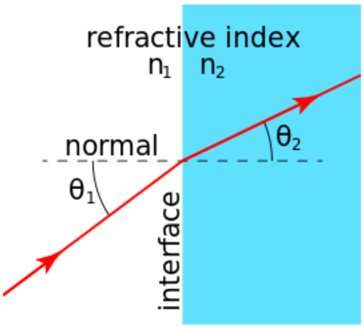

Figure 1: Refracted Light

In this figure, light is depicted as moving from one medium with a *refractive index* of n_1 into a different medium at an *angle of incidence* θ_1. The new medium has a (potentially different) refractive index of n_2 and thus the new *angle of refraction* is θ_2.

This is all ruled by Snell's Law which states that

$$n_1 \sin\theta_1 = n_2 \sin\theta_2$$

Refractive indexes of materials are computed as a ratio of the speed of light in a vacuum and the phaser velocity of light in the medium ($n = \frac{c}{v}$). For example, the index of refraction of water is typically 1.333 which means that light travels 1.333 times faster in a vacuum than in water. Air (at one atmosphere) essentially has an index refraction of 1 (essentially the same as a vacuum).

Write a program that computes the angle of refraction, θ_2 given the angle of incidence, θ_1 and the index of refraction of the medium, n_2. We'll assume that the first medium is always air and so $n_1 = 1$.

Your program will prompt the user for θ_1 (in degrees) and n_2 and do basic error checking, rejecting any invalid values. It will then compute the angle of refraction using Snell's Law. Your output may look something like the following.

```
Angle of incidence: 45 degrees
Index of refraction: 1.333
Angle of refraction: 32.04 degrees
```

2. The rates of return on an investment are usually expressed in terms of an annualized return rate (or *Compound Annual Growth Rate*). Suppose that an investment has an initial value of v_0. After a certain number of years, t, suppose that the investment has a value of v_1. To compute the annualized return rate, you can use the formula

$$\left(\frac{v_1}{v_0}\right)^{\left(\frac{1}{t}\right)} - 1$$

For example, suppose that an investment is made with an initial value of \$200. Then 5 years, 7 weeks and 3 days later (that is, ≈ 5.14256 years later, ignore leap year days for this exercise), it has a value of \$275. That is, the investment has grown by $\frac{275-200}{200} = 37.5\%$, but that was over more than 5 years. Annually, the rate of growth would be:

$$\left(\frac{275}{200}\right)^{\left(\frac{1}{5.14256}\right)} - 1 = 6.388\%$$

Write a program that prompts the user for the initial and final value of the investment as well as the number of years, weeks, and days between the initial and final values and then computes and outputs the annualized rate of return. Your output may look something like the following.

```
Initial Value: $200.00
Final Value:   $275.00
Years: 5.14256
Annualized Rate of Return: 6.388%
```

3. In this exercise you will get some familiarity with debugging, troubleshooting, software testing and git, a distributed version control system, all essential tools of software engineering and development.

In particular, you'll be using GitHub (`https://github.com/`) as your git system. Before you begin, you will need to create an account on GitHub and read the course tutorial on getting started with git: `http://cse.unl.edu/~cbourke/gitTutorial.pdf`. Alternatively, you can find your own resources (and share them on Piazza). When you sign up on GitHub be sure to use a `.edu` email address and go through the process to get a free student account so that you can create private repositories. You are *highly encouraged* to start using git/GitHub (or something similar) for all of your future assignments. However, be sure to commit code to a *private* repository so that you do not publicly share code which would violate the course and department's academic integrity policy. For this exercise, however, keep your repositories public so that we can grade your assignment. You are being graded more on the git process rather than the code itself.

Gomer has written a solution to his CS1 course's first assignment which involves computing whether or not a given integer is a *Kaprekar Number* (details can be found in his documentation). However, he has committed his code with a lot of errors, both syntax errors and logic errors. Fortunately Gomer is actually pretty good at writing test cases to check that his code actually works. He has written both Java and C versions (you only need to concern yourself with the C version for this course) of his program and has provided several ways of testing his code using an ad-hoc testing suite as well as a more formal testing framework. The project is available at:

`https://github.com/cbourke/KaprekarProject`

Fork this project so that you have your own copy on GitHub. *Then clone* the forked project to your local development environment so that you can work with it. Then:

a) Fix all the syntax errors so that you get a working executable, commit and push your results.

b) Run the test suite(s) and fix all the logic errors so that all the test cases execute fully. **For each change/fix, make an independent commit/push** with specific commit comments so that a complete history will be available of all the bugs.

Further instructions for building and running the project are available in the project's `readme.md` file.

What you need to handin: Instead of handing your program in, we'll want to verify that you've followed the process by checking your repository and its commit history. Hand in a text file named `readme.md` with the URL of your repository.

Additional Resources

- Pro Git, free online book: `https://git-scm.com/book/en/v2`
- GitHub Desktop Client: `https://desktop.github.com/` (if you don't want to work from the command line)