Student Housing

Branches and Loops

Assignment credit: adapted from Evan Peck's Housing Algorithms

The goal of this assignment is to practice designing algorithms, drawing flowcharts, and applying your knowledge of branches and loops to solve a problem. You are also encouraged to think about how designing algorithms can have social impacts in the real world, and to think beyond "does it work?" to "how should it work?"

You are encouraged to work with a partner (or two) if desired.

You are welcome to use your textbook as a reference, but do not use general internet searches or large language models (LLM) like chat gpt.

Part 0: Planning

We're going to design an application to determine what order students get to choose their housing. We're going to use a point system. The idea is that the application will ask a student a series of questions, and use the answers to generate a score. The higher the score, the sooner the student gets to choose their housing.

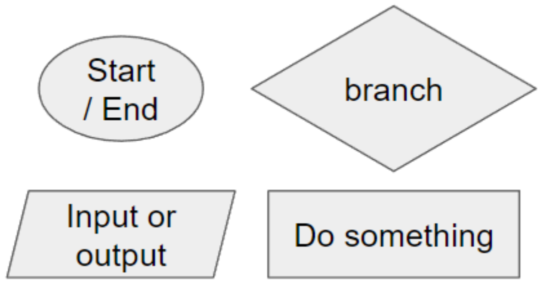
We need to come up with a list of factors that should be considered, such as age, class year (senior, junior, etc.), and so forth. You may also want to think about fairness, and include factors that take into account the diverse needs of students. Write down at least 6 possible factors. Make sure at least 2 of them are numeric. Decide how each factor should relate to a final point value. For example, maybe being a senior is worth 10 points, and being a junior is worth 6 points, etc. You may include negative factors, too. Possibly a suspension would be worth negative points.

Next we'll design our algorithm on paper as a **flowchart**. The algorithm should:

1. Ask students questions (e.g., *what class are you?*)
2. Include at least 4 questions.
3. Assign points based on their answers (e.g., 10 points for a senior)
4. Accumulate the total points across all the answers (e.g., *you have 23 housing points*)
5. Include at least one AND or OR
6. Include at least one question that only appears if the previous question was answered in a specific way. For example, maybe only 4th year students would be asked whether they are about to graduate.

7. Include at least one question that uses math. For example, you might divide the number of credits by 10 to get points.

Draw a flowchart on paper showing how these factors will be used to calculate a final score for a student. Use the four kinds of shapes shown below:



Discuss your flowchart with another student or two to make sure it makes sense and make modifications as needed.

Part 1: Test Cases

Write 3-5 *test cases* for your algorithm. Test cases are examples of inputs and the desired output. You should write these before writing your code (in the next step), and then check that the code works as intended according to the test cases after it's written.

A test case might be something like: a 25-year-old senior who is on academic probation should output 10 points.

Part 2: Implementation

Now it's time to translate your algorithm to Python. Create a new Python script. Make sure to include a block comment at the top of the script with the purpose of the file, your name, and the date.

Working from your flowchart, write the logic in Python. Try writing just a few lines and then running your program frequently as you go to make sure things are working as intended.

Part 3: Unit Testing

Unit testing is the process of checking your code against the test cases you wrote before coding. Make sure the given inputs produce the correct outputs.

Part 4: Loop

Next enhance your program to allow it to work for multiple students instead of just one. Add a prompt at the beginning that says: **Enter the next student’s name or type q to quit**. Add a loop so that your code will run each time another student’s name is entered and quit when q is entered.

Part 5: User Testing

In real-world software development, there are generally multiple phases of testing. Unit testing is when a developer tests their own program. User testing is when (often non-technical) users test the program. Simulate user testing by getting a classmate to run your program and give you feedback.

Part 6: Completion

Congratulations on reaching the end of the assignment. The final step is to reflect on what you’ve learned. Consider the following questions on your own; we will discuss them as a class next time.

- What was the most challenging part?
- How helpful do you think it would have been to have used a LLM to help?
- How fair is your algorithm? Does it advantage some groups of students over others?