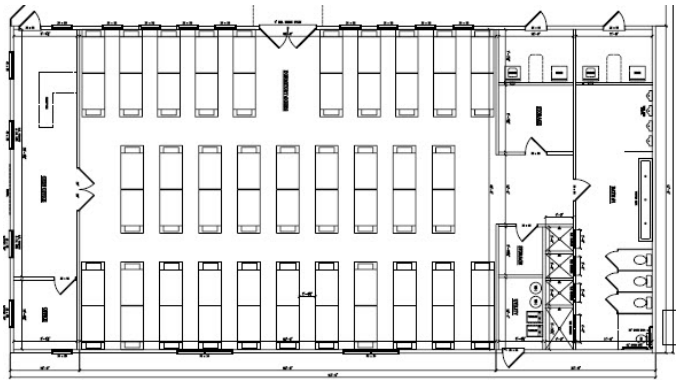
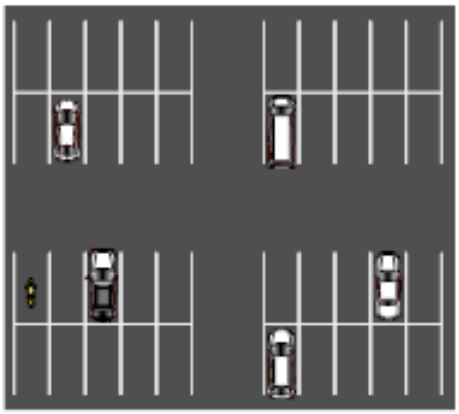
CSCI561 Fall 2018 Foundations of Artificial Intelligence

Homework 2 VERSION 2.0

Due October 22, 2018 23:59:59



LAHSA Shelter



SPLA Parking Lot

Problem Description:

Los Angeles Homeless Services Authority (LAHSA) and Safe Parking LA (SPLA) are two organizations in Los Angeles that service the homeless community. LAHSA provides beds in shelters and SPLA manages spaces in parking lots for people living in their cars. In the city's new app for homelessness, people in need of housing can apply for a space with either service. For this homework, you will help SPLA choose applicants that meet the SPLA specific requirements for the space and that also optimize the use of the parking lot for that week.

Applicant information entered into the homelessness app:
Applicant ID: 5 digits
Gender: M/F/O
Age: 0-100
Pets: Y/N
Medical conditions: Y/N
Car: Y/N
Driver's License: Y/N
Days of the week needed: 0/1 for each day of the 7 days of the week (Monday-Sunday)

Example applicant record: 00001F020NNYY1001000 for applicant id 00001, female, 20 years old, no pets, no medical conditions, with car and driver's license, who needs housing for Monday and Thursday.

SPLA requirements differ from LAHSA. They are both picking from the same applicant list. They each have different resources and may not be qualified to accept the same applicants. SPLA and LAHSA alternate choosing applicants one by one. They must choose an applicant if there is still a qualified one on the list (no passing). SPLA applicants must have a car and driver's license, but no medical conditions. LAHSA shelter can only serve women over 17 years old without pets. Both SPLA and LAHSA have limited resources that must be used efficiently. Efficiency is calculated by how many of the spaces are used during the week. For example, a SPLA parking lot has 10 spaces and can have at most 10*7 days = 70 different applicants for the week. SPLA tries to maximize its efficiency rate.

Input: The file `input.txt` in the current directory of your program will be formatted as follows:

First line: strictly positive 32-bit integer ***b***, number of beds in the shelter, ***b*** <= 40.

Second line: strictly positive 32-bit integer ***p***, the number of spaces in the parking lot

Third line: strictly positive 32-bit integer ***L,*** *number of applicants chosen by LAHSA so far.*

Next ***L*** lines: **L** number of Applicant ID (5 digits) , separated with the End-of-line character LF.

Next line: strictly positive 32-bit integer ***S***, *number of applicants chosen by SPLA so far.*

Next ***S*** lines: **S** number of Applicant ID (5 digits), separated with the End-of-line character LF.

Next line: strictly positive 32-bit integer ***A,*** *total number of applicants*

Next ***A*** lines: the list of **A** applicant information, separated with the End-of-line character LF.

Output:

Next applicant chosen by SPLA: Applicant ID (5 digits)

Input.txt

```
10
10
1
00005
1
00002
5
00001F020NNYY1001000
00002F020NNYY1000111
00003M040NNYY1000110
00004M033NNYY1000000
00005F020NNYY1000110
```

Output.txt

```
00001
```

Explanation

SPLA chose 00002 first, and then LAHSA chose 00005 next. 00001 should be chosen because it allows SPLA to choose applicants 00001, 00003, and 00004, while LAHSA doesn't need to choose anybody else. This is because 00001 qualifies for both, but 00003 and 00004 only qualify for SPLA. So, choosing 00001 next means that SPLA chooses all three remaining applicants, and LAHSA gets zero.

Guidelines

This is a programming assignment. You are provided sample input and output files. Please understand that the goal of these samples is to check that you can correctly parse the problem definitions, and generate a correctly formatted output. The samples are very simple and it should not be assumed that if your program works on the samples it will work on all test cases. There will be more complex test cases and it is ***your task to make sure that your program will work correctly on any valid input***. You are encouraged to try your own test cases to check how your program would behave in some complex special case that you might think of. Since **each homework is checked via an automated A.I. script**, your output should match the specified format *exactly*. Failure to do so will most certainly cost points. The output format is simple and examples are provided. You should upload and test your code on vocareum.com, and you will submit it there.

Grading

Your code will be tested as follows: Your program must not require any command-line argument. It should read a text file called "input.txt" in the current directory that contains a problem definition. It should write a file "output.txt" with your solution to the same current directory. Format for input.txt and output.txt is specified below. End-of-line character is LF (since Vocareum is a Unix system and follows the Unix convention).

The grading A.I. script will

- Create an input.txt file, delete any old output.txt file.
- Run your code.
- Test your output.txt file

Academic Honesty and Integrity

All homework material is checked vigorously for dishonesty using several methods. All detected violations of academic honesty are forwarded to the Office of Student Judicial Affairs. To be safe you are urged to err on the side of caution. Do not copy work from another student or off the web. Keep in mind that sanctions for dishonesty are reflected in *your permanent record* and can negatively impact your future success. As a general guide:

- **Do not copy** code or written material from another student. Even single lines of code should not be copied.
  Do not collaborate on this assignment. The assignment is to be solved individually.
  Do not copy code off the web. This is easier to detect than you may think.
- **Do not share** any custom test cases you may create to check your program's behavior in more complex scenarios than the simplistic ones considered below.
  Do not copy code from past students. We keep copies of past work to check for this.
- **Do** ask the professor or TA if you are unsure about whether certain actions constitute dishonesty. It is better to be safe than sorry.

Homework Rules

1. Use Python 2.7 to implement your homework assignment. You are allowed to use standard libraries only. You have to implement any other functions or methods by yourself.

2. Create a file named "hw2cs561f2018.py". When you submit the homework on labs.vocareum.com, the following commands will be executed:

   python hw2cs561f2018.py

3. Create a file named "output.txt" and print its output there. For each test case, the grading script will put an "input.txt" file in your work folder, runs your program (which reads "input.txt"), and check the "output.txt" file generated by your code. The grading script will replace the files automatically, so you do NOT need to do anything for that part.

4. Homework must be submitted through Vocareum. Please only upload your code to the "/work" directory. **Don't create any subfolder or upload any other files**. Please refer to http://help.vocareum.com/article/30-getting-started-students to get started with Vocareum.

5. Your program must handle all test cases within a maximum runtime of **3 minutes** per test case on Vocareum.

6. It is recommended to submit your program 24 hours ahead of the deadline to avoid any submission issues on Vocareum. Late submissions will not be graded.

Helpful Hints:

1. You must accept an applicant for every day they request.
2. Only women can be accepted into LAHSA.
3. In the case of a tie, choose the smaller applicant ID.
4. When one agency runs out of applicants to choose, the agencies are done choosing. The agency with remaining applicant choices can take all remaining applicants that qualify
5. **We will not give unsolvable inputs.** This means that we won't give any irregular inputs that don't conform to the format we've described in this document.
6. **Some inputs may have more than one valid solution.** We will accept all solutions that achieve the maximum possible efficiency for the given input.
7. **Think about representing the problem.**
    a. What is a good representation of states and operators?
    b. How can you use this to simplify the problem representation?
    c. How will you evaluate the "score" of a state?
8. **Think about complexity.**
    a. What affects branching factor?
    b. What affect search tree depth?
    c. How can you use the input parameters to determine which algorithm will be able to generate a solution within 3 minutes?
9. *https://news.vice.com/en_us/article/qvmgem/california-has-a-hidden-homelessness-crisis*