Artificial Intelligence

Machine Problem 3 – Course Planning using a Constraint Satisfaction Problem Formulation

Introduction

For this assignment, you will implement a program that uses a Constraint Satisfaction Problem (CSP) formulation to find possible degree plans for students in the M.S. in Data Science program at Lewis University. A *degree plan* is a mapping of academic terms to courses. For example, “Year 1 Fall 2” to “MATH-51100”. This information can be used to help students pick courses and also to determine a course rotation that allows students to complete the degrees in the specified number of terms.

Requirements

You are to use Python 3 with the python-constraint package to determine the number of possible degree plans for a given start and end term under the constraints given below. Also, your code should generate one possible degree plan that satisfies all constraints. The information needed for formulating the CSP is given within the two sheets of the csp_course_rotations.xlsx file. The first sheet (course_rotations) provides a listing of all available courses, their type (foundation, core, elective, or capstone), and the term availability (0 - unavailable, 1 - available) for each of the terms (1: Fall 1, 2: Fall 2, 3: Spring 1, 4: Spring 2, 5: Summer 1, 6: Summer 2). The second sheet (prereqs) specifies which course must be taken before another. Note that a course may have multiple prerequisites.

Your program must output the number of possible degree plans and one possible degree plan for a student that starts in Year 1 Fall 1 and finishes in Year 3 Fall 2. The degree plan must satisfy the following requirements:

1. Student will take one and only one course per term.
2. Course that has prerequisites must be taken in a term that follows the term in which all prerequisites are done.
3. The student does not need to repeat courses.
4. Some terms may be skipped as long as the student finishes in Year 3 Fall 2.
5. Student needs to take 3 out of the 8 elective courses. It doesn’t matter which ones are included in the degree plan. Those courses which are not taken will be labeled as “Not Taken” (see sample output).
6. Student must take all foundation and core courses.

The program will generate the output shown in the sample output at the end.

Additional Requirements

1. The name of your source code file should be `mp3.py`. All your code should be within a single file.
2. You can only import numpy, pandas, and constraint packages.
3. Your code should follow good coding practices, including good use of whitespace and use of both inline and block comments.
4. You need to use meaningful identifier names that conform to standard naming conventions.
5. At the top of each file, you need to put in a block comment with the following information: your name, date, course name, semester, and assignment name.

What to Turn In

You will turn in the single mp3.py file using BlackBoard.

HINT

- You can load an Excel file using pandas’ pd.read_excel function. The sheet is specified by the sheet_name attribute.

Grading Rubric

Category	Unsatisfactory (0-1 points)	Satisfactory (2-3 point)	Distinguished (4-5 points)
Program Correctness	• Program does not execute due to errors • Incorrect results for most or all input	• Program works and completes most tasks appropriately • Program fails to work for special cases	• Program runs and completes all required tasks • Handles any required special cases • Executes without errors
Programming Style	• No name, date, or assignment title included • Poor use of white space • Disorganized and messy • No or few comments in the source code • Poor use of variables (improper scope/visibility, ambiguous naming).	• Includes name, date, and assignment title. • White space makes program fairly easy to read. • Well organized code. • Some comments missing in the source code or too many comments • Good use of variables (few issues with scope/visibility or unambiguous naming).	• Includes name, date, and assignment title. • Excellent use of white space. • Perfectly organized code. • Source code is commented throughout when needed • Excellent use of variables (no issues with scope/visibility or unambiguous naming).
Following Specifications	• Incorrect filenames • Incorrect specified identifier names • Source code organization different from requirements • Additional requirements not satisfied	• Correct filenames and class names • Few issues with other specified identifier names • Source code organization close to requirements • Some additional requirements not satisfied	• Correct filenames and specified identifier names • Source code organization satisfies all requirements • All additional requirements satisfied

Sample Program Output

```
CLASS: Artificial Intelligence, Lewis University
NAME: [put your name here]

START TERM = Year 1 Fall 1
Number of Possible Degree Plans is 9488

Sample Degree Plan
Not Taken          CPSC-57400
Not Taken          CPSC-57200
Not Taken          CPSC-57100
Not Taken          CPSC-55200
Not Taken          CPSC-51700
Year 1 Fall 1      CPSC-50600
Year 1 Fall 2      MATH-51100
Year 1 Spring 1    MATH-51000
Year 1 Spring 2    MATH-51200
Year 1 Summer 1    CPSC-50100
Year 2 Fall 1      CPSC-51100
Year 2 Fall 2      CPSC-53000
Year 2 Spring 1    CPSC-54000
Year 2 Spring 2    CPSC-55500
Year 2 Summer 1    CPSC-51000
Year 2 Summer 2    CPSC-52500
Year 3 Fall 1      CPSC-55000
Year 3 Fall 2      CPSC-59000
```