CSIT 552 HW1
Topic: Python Basics

Dr. Boxiang Dong

1 Problem Description

Instructions. Please write code in a Python notebook to complete the following tasks.

- Task 1 (10 pts). Define a *Student* class with the following attributes:
  - name: str
  - cwid: an array of 8 digits (like your MSU id, it has 8 digits after it)
  - grades: an array of 10 numerical grades in the set of (4.0, 3.0, 2.0, 1.0 and np.nan).

  The constructor takes the name as argument and generates cwid and grades by calling the functions implemented in Task 2 and 3.

- Task 2 (10 pts). Implement a private function named *generate_cwid* that generates a random cwid. Note that the first digit in a cwid should not be 0. For example, "12345678" is a valid cwid, but "00123456" is not. To generate a random cwid, you cannot use *itertools.combinations_with_replacement()*. Instead, you should use the random functions provided in NumPy.

- Task 3 (15 pts). Implement a private *generator* function named *generate_grades* that returns an array of random grades. Note that the array size should be 10. Also, the array may include np.nan (which stands for None in NumPy). All np.nan must stay at the end of the array. For example, [1.0, 2.0, 3.0, 4.0, 1.0, 2.0, 3.0, 4.0, np.nan, np.nan] is a valid grade array, but [1.0, 2.0, 3.0, 4.0, 1.0, np.nan, 3.0, 4.0, np.nan, np.nan] is not.

- Task 4 (10 pts). Implement the magic str function to return the string representation of a student in the format of *name (cwid): grades*. Note, in the return value, there should not be any np.nan in the grades. For example, Ashley Young (12345667): [1.0, 2.0, 3.0, 4.0, 1.0, 2.0, 3.0, 4.0]. Here, only 8 grades are returned in the str because the last 2 grades are np.nan.

- Task 5 (10 pts). Outside the Student class, implement a function named *simulate_students* that generate 100 students (you are free to provide the names) and return them in a list.
- Task 6 (15 pts). Outside the Student class, implement a function named *calculate_gpa* that takes in a list of students and returns an array of their gpas. Note that when calculating gpa, assuming that each course has 3 credits. Also, exclude np.nan in the gpa calculation. For example, if the grades for a student is [1.0, 2.0, 3.0, 4.0, 1.0, 2.0, 3.0, 4.0, np.nan, np.nan], the GPA should be 2.5, rather than 2.0. Note that you should use NumPy ufunc instead of Python's native aggregation functions whenever possible.
- Task 7 (15 pts). Outside the Student class, implement a function named *find_best_students* that takes in a list of students and returns nothing. This function should find 3 students who have taken at least 6 courses and have the highest gpa, and print their string representation. These 3 students do not have to be ordered by their gpas. Note that you should use NumPy ufunc instead of Python's native aggregation functions whenever possible.
- Task 8 (15 pts). Outside the Student class, implement a function named *find_risky_students* that takes in a list of students and returns nothing. This function should find all students who have completed at least 2 courses with a grade of 2.0 or less, and print their string representation. These students should be ordered by the number of courses with a grade of 2.0 or less, from the highest to the lowest. Note that you should use NumPy ufunc instead of Python's native aggregation functions whenever possible.

2 Submission Guideline

1. Work individually.
2. Please submit a .ipynb file.
3. Submit your solution on Canvas on time. A late penalty of 10 points for each late day applies. Any late for more than three days receives zero automatically.