CSC 6301 final exam Fall 1 2025

Important:

You have **four hours** to solve all eight questions.

You have to submit **one single pdf** with your answers. In this pdf you can put electronic edited text, photo of handwritten text or annotations, print screen images, or any other materials you believe to be appropriate as your answer.

Remember to number your answers from 1 to 8.

If you have any issues submitting your pdf file, please send an email immediately to **krishnamurtg@merrimack.edu** stating your issue and attaching your pdf file with answers.

1) Include the docstrings documentation

Given the code below, insert the docstrings to properly document the class and all its methods.

You will need to explain the code considering that the documentation reader will need to understand the code enough to provide maintenance.

```
1   class Queue:
2       def __init__(self):
3           self.a_in = []
4           self.a_out = []
5       def enqueue(self, d):
6           self.a_in.append(d)
7       def dequeue(self):
8           if (self.a_out == []):
9               for d in self.a_in:
10                  self.a_out.append(d)
11              self.a_in = []
12          return self.a_out.pop(0)
```

For ease, here below is the code in the textual version if you want to copy it in your editor.

```
class Queue:
    def __init__(self):
        self.a_in = []
        self.a_out = []
    def enqueue(self, d):
        self.a_in.append(d)
    def dequeue(self):
        if (self.a_out == []):
            for d in self.a_in:
                self.a_out.append(d)
            self.a_in = [] return
        self.a_out.pop(0)
```

2) **Convert from python to Java**

Convert the code below from Python to Java and follow all best practices to Java code including choice of Identifiers and Code Formatting.

```
def RSP():
    guess = ""
    while (True):
        guess = input("Enter (r)ock, (s)cissors, or (p)aper: ")
        if (guess not in ["r", "s", "p"]):
            print("Only 'r', 's', or 'p' are valid inputs! Please try again.")
        else:
            break
    rand = random()
    if (rand < 1/3):
        comp = "r"
    elif (rand < 2/3):
        comp = "s"
    else:
        comp = "p"
    if (guess == comp):
        print("It is a tie!")
    elif ((guess == 'r') and (comp == 'p')) or \
         ((guess == 'p') and (comp == 's')) or \
         ((guess == 's') and (comp == 'r')):
        print("Sorry, you lost as I had", comp)
    else:
        print("Congrats, you won as I had", comp)

RSP()
```

3) **Class Diagram**

Draw the class diagram for the Java code represented by the following classes.

```java
public class Person {
    String name;
    int age;
    void display() {
        //method body;
    }
}

public class Student extends Person {
    String major;
}

public class Instructor extends Person {
    String title;
}

public class Course {
    String name;
    int credits;
    Instructor teacher;
    Student roaster[];
    void addStudent(Student name) {
        // method body
    }
    void display() {
        // method body
    }
}

public class OnlineCourse extends Course {
    String meetingId;
    Person helper;
}
```

4) **Java Collections**

Using your words, explain the structural and Java implementation differences between the Java Collections `Stack` and `Queue`. I suggest you limit your answer to no less than 2 lines and to no more than 10 lines (soft limits).

5) SDLC

Using your words, what is the advantage of having different teams to develop code and to test it? I suggest you limit your answer to no less than 2 lines and to no more than 10 lines (soft limits).

6) Version Control

How can we recreate a reusing environment for a Python program that needs packages numpy version 1.18.5 and xlwt version 1.3.0 for running your own code? What command line command is necessary? Do you need additional files? Do you need to make changes in the .py files? I suggest you limit your answer to no less than 2 lines and to no more than 10 lines (soft limits).

7) Profiling

Considering the code below and the obtained output, which function would you consider optimizing? Explain your choice. I suggest you limit your answer to no less than 2 lines and to no more than 10 lines (soft limits).

```python
from itertools import combinations

def permutations(n):
    ones = list(combinations(list(range(n)),n//2))
    ans = []
    for o in ones:
        case = []
        for i in range(n):
            if (i in o):
                case.append(1)
            else:
                case.append(0)
        ans.append(case)
    return ans

def check(mat):
    n = len(mat[0])
    for j in range(n):
        acc0, acc1 = 0, 0
        for i in range(len(mat)):
            if (mat[i][j] == 1):
                acc1 += 1
            elif (mat[i][j] == 0):
                acc0 += 1
            if (acc0 > (n//2)) or (acc1 > n//2):
                return False
    return True

def layer(r, mat, perm, ans):
    for p in perm:
        mat.append(p)
        if check(mat):
            if (r+1 == len(p)):
                ans += 1
            else:
                ans = layer(r+1, mat, perm, ans)
        mat.pop()
    return ans

def balanced01mat(n):
    perm = permutations(n)
    ans = layer(0, [], perm, 0)
    return ans

import cProfile
cProfile.run('print("Balanced matrices of order 6 is", balanced01mat(6))')
```

```
Balanced matrices of order 6 is 297200
         49582063 function calls (49198183 primitive calls) in 39.456 seconds

   Ordered by: standard name

   ncalls  tottime  percall  cumtime  percall filename:lineno(function)
        1    0.000    0.000   39.456   39.456 <string>:1(<module>)
  7677620   31.690    0.000   33.616    0.000 balanced01mat.py:16(check)
 383881/1    4.361    0.000   39.456   39.456 balanced01mat.py:29(layer)
        1    0.000    0.000    0.000    0.000 balanced01mat.py:3(permutations)
        1    0.000    0.000   39.456   39.456 balanced01mat.py:40(balanced01mat)
        1    0.000    0.000   39.456   39.456 {built-in method builtins.exec}
 26165176    1.979    0.000    1.979    0.000 {built-in method builtins.len}
        1    0.000    0.000    0.000    0.000 {built-in method builtins.print}
  7677760    0.705    0.000    0.705    0.000 {method 'append' of 'list' objects}
        1    0.000    0.000    0.000    0.000 {method 'disable' of '_lsprof.Profiler' objects}
  7677620    0.721    0.000    0.721    0.000 {method 'pop' of 'list' objects}
```

8) SDM

Which among the methodologies Agile, Lean, and DevOps is more similar (or less different) to the Waterfall methodology? Explain your choice. I suggest you limit your answer to no less than 2 lines and to no more than 10 lines (soft limits).