Design and Analysis of Software Systems **Final Exam – Spring 2024**

INSTRUCTIONS: This is a closed-book, closed-notes exam. You may use a one page 8 ½ * 11 inch hand-written cheat sheet. Xerox copies or any downloaded notes/slides is not allowed.

It consists of 18 questions. You are to answer all questions. You have 180 minutes to complete the exam for a total of 70 points.
Credit is given for what you write, not what you are thinking. Partial credit will be given based on Quality of the content, not Quantity. A voluminous content-free answer may simply annoy the grader.

---

1. **(1 point)** Which of these are true for non functional requirements?
   a) A non-functional requirement is also called behavioral requirements
   b) A non-functional requirement is a statement that a software product must have certain properties
   c) It consists of Development and operational requirements
   d) All of the mentioned

2. **(1 point)** Which of the following statement is true?
   a) The job of creating, modifying, and managing requirements over a product's lifetime is called requirement development
   b) The portion of requirements engineering concerned with initially establishing requirements is termed requirements engineering
   c) The portion of requirements engineering concerned with controlling requirements changes is called requirement management
   d) All of the mentioned.

3. **(1 point)** 58000 LOC gaming software is developed with effort of 3 person-year. What is the productivity of person-month?
   (a) 1.9 KLOC
   (b) 1.6 KLOC
   (c) 4.8 KLOC
   (d) 4.2 KLOC

4. **(1 point)** In a class definition with 10 methods, to make the class maximally cohesive, number of direct and indirect connections required among the methods are
   (a) 90, 0
   (b) 45, 0
   (c) 10, 10
   (d) 45, 45

5. **(1 point)** Alpha and Beta testing are forms of
   (a) Black box testing
   (b) White box testing
   (c) Acceptance testing
   (d) System testing

6. **(2 point)** Cohesion is a qualitative indication of ?

7. **(2 points)** Do you agree with this statement - "Exhaustive testing is possible and should be the goal for all non-trivial software applications." Why or why not?

8. **(3 points)**
Your company has just bought a math software package that solves nonlinear simultaneous equations, and you have been asked to test it thoroughly. You have documentation that explains the functionality of the package, but no source code. Of the following types of coverage, which types of coverage would you aim for: *statement coverage, function coverage, requirements coverage, path coverage.* Give reasons for your answer.

9. **(3 points)**
In almost all instances, you should "break" your software Design into many modules, hoping to make understanding easier and as a consequence, reduce the cost required to build the software. Describe the trade-off between cost of software and pursuing higher number of modules using a graph.

10. **(3 points)** How does a sequence diagram differ from a state diagram? How are they similar?

11. **(3 points)**
You are working in a team of 10 people to build a math library product that contains over 150 mathematical operations, such as solving simultaneous equations, compute Fourier and Laplace transforms, solving differential equations, perform numerical integration etc. Each member of the team is working on a few of these routines. There is some common code used by all the routines to perform utility functions such as sorting values, and a Façade class that provides the external interface to the library. What type of integration approach would you choose for this application, and why?

12. **(3 points)**
Identify the following types of integration strategies:
(a) Taking the entire integrated system and testing it for the first time as a unit
(b) Start by testing only the user interface with the underlying functionality simulated by stubs:
(c) Testing the lower levels of the system via drivers

13. **(6 points)**
A not-very-user-friendly developer is creating software for a library, to help patrons find books. Their user interface design as described below. The text in italics is what the program displays to the user. The text in parentheses is our description of how the user interface works.

Step 1: *Enter the type of search ("title", "subject", "author"):* _____

(The dashes actually represent a text box into which the user is expected to type in one of the words. If they enter a value other than these three, they get an error message).

Step 2: *Enter whitespace-separated search terms. Please note that the search is case-sensitive and the relative ordering is used in determining match scores.* _____

(After the user enters the words to search for, the results are displayed. If no matches are found, the following happens)

Step 3: (A dialog box pops up): "*Error 203: Bad match. Fix and try again. Press OK to continue*".

(The dialog box has only an OK button).

Step 4: (If they press OK, then the next dialog box pops up.) "*Do you want to do another search?*"):

(There are three buttons on this dialog box: "Yes", "No", "Cancel". If the user pushes "Yes", they are taken back to Step 1. If they push "No" or "Cancel", they are taken back to Step 1).

Identify four *distinct* problems with this user interface design and describe an appropriate solution to *each* of these problems.

14. (6 points)

Identify ONE behavioral design pattern that can be appropriately used common across the three applications: Stock Market Updates. Temperature Monitoring. Chat application. Please explain clearly how the pattern can be applied in the context of any one of the applications. Use a class diagram to show the appropriate

You may use the template table given below (add lines if necessary):

Generic GoF Pattern Name:		
Participants (application specific):		
Class	**Participant**	**Role of participant (as per the pattern)**

15. (6 points)

Draw a state machine for the following system.

A clock has two modes of operation: *normal* and *time-set*. In *normal* mode it displays the time. When the *set* button is pressed the clock starts flashing the time and enters *time-set* mode. Every time the *advance* button is pressed while in *time-set* mode the clock increments the time. When the *set* button is pressed while the clock is in *time-set* mode the time display stops flashing and the clock returns to *normal* mode. The clock starts in *normal* mode.

16. (6 points) Inspect the code and identify any potential issues, inefficiencies, or areas for improvement.

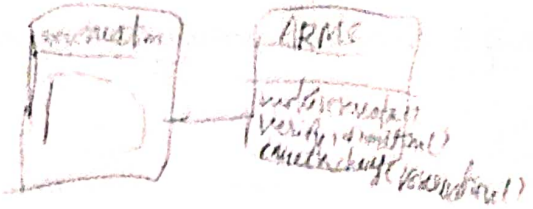
```
def mystery_function(n):
    if n == 0:
        return 0
    elif n == 1:
        return 1
    else:
        return mystery_function(n-1) + mystery_function(n-2)

def is_prime(num):
    if num <= 1:
        return False
    elif num <= 3:
        return True
    elif num % 2 == 0 or num % 3 == 0:
        return False
    i = 5
    while i * i <= num:
        if num % i == 0 or num % (i + 2) == 0:
            return False
        i += 6
    return True

def generate_fibonacci_primes(limit):
    fibonacci_primes = []
    for i in range(limit):
        fib_number = mystery_function(i)
```

```
    if is_prime(fib_number):
        fibonacci_primes.append(fib_number)
return fibonacci_primes

print(generate_fibonacci_primes(10))
```



17. (Three parts, 12 points total)

Develop an automated airline reservations system (ARS) for Fly-By-Night Airlines. Passengers can make reservations at automated reservation machines (ARMs) that are located throughout cities served by Fly-By-Night. A reservation will include the flight number, departure date and time, reservation type (first class, coach), a seat number, and the price of the ticket. Once reservations are made and credit card information is verified, tickets are printed for the passenger. Passengers can also cancel or change reservations after they are made. All flight data is entered by a designated data entry clerk.

Note: You may make simple assumptions about this problem, based on your own knowledge about airline reservation systems.

a) **(3 points)** Write the detailed textual use case (step-by-step) for a passenger successfully using the ARM to book a reservation.

b) **(5 points)** Draw a UML Design class diagram for the ARS showing the classes, relationships, and multiplicities between classes

c) **(4 points)** Basis your class diagram, create a UML sequence diagram for a passenger successfully booking a reservation.

18. (10 points)

SoftModel Inc. has been contracted to build a system for parking garages attached to the Space Stations of the 22nd century. The first parking garage to be built will be attached to "Space Station Siren" which orbits Titan, Saturn's moon. The parking garage will be able to park up to 100 space vehicles of all types. It will be have 3 air locks which allow the space vehicles to move from airless space into the pressurized atmosphere of the space station. Each air lock can fit either one space ore vehicle or 3 regular (passenger or military) space vehicle. Future parking garages may have more (or fewer) parking spaces and a different number and size air lock.

The parking technology is similar to those currently used on Earth. Drivers can enter the garage with a permit that can be swiped with a card reader. They can also get a ticket by pressing a button at the entrance. The vehicles have automatic arms to get the ticket inside. You don't want to open a window and let the air out. The ticket has a date and time-stamp indicating the date/time of entering the garage. After swiping the permit or getting a ticket, the driver enters an air lock (if there is room) and then enters the garage itself.

The system to be developed by SoftModel should keep track of the number of cars currently in the garage, and in each air lock. It displays signs indicating whether or not the garage is full, and whether or not each air lock is full. Drivers with tickets pay the attendant at a gate before leaving the garage; drivers with permits have a parking fee added to their account when they swipe their permits upon exiting from the garage. Drivers must pass through an air lock to exit the garage. An operator console displays the status of the system including the number and types of (1) parked vehicles, (2) vehicles in air locks, and (3) vehicles waiting for space in the lot or space in the air locks.

The system must keep track of the payments being made, and the amounts due from permit holders. It must also provide summaries to authorized personnel regarding peak garage hours and the number of permits used.

Design a single **state diagram** for the above system. Please state your assumptions clearly (if any).