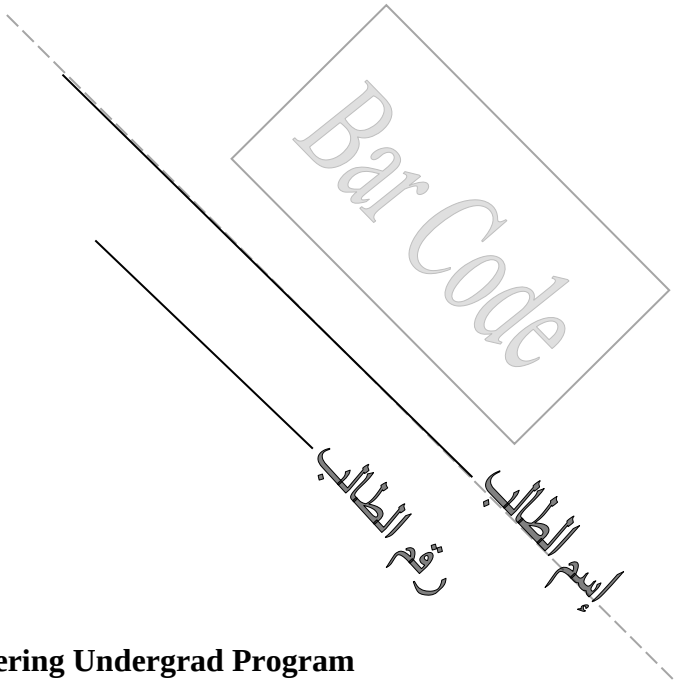

Cairo University
Faculty of Computers and Artificial Intelligence

Final Exam

Department: Computer Science – Software Engineering Undergrad Program

Course Title: Software Testing
Course Code: SCS 252
Semester: Winter 2021
Instructor: Dr Soha Makady
Date: Jul. 4th, 2021
Exam Duration: 2 Hours

تعليمات هامة

- حيازة التيلفون المحمول مفتوحا داخل لجنة الأمتحان يعتبر حالة غش تستوجب العقاب وإذا كان ضرورى الدخول بالمحمول فيوضع مغلق فى الحقائب.
- لا يسمح بدخول سماعة الأذن أو البلوتوث.
- لايسمح بدخول أي كتب أو ملازم أو أوراق داخل اللجنة والمخالفة تعتبر حالة غش.

Question	Mark	Signature
One		
Two		
Three		
Four		
Five		
Six		
Seven		
Eight		
Nine		
Ten		
Total Marks		

_____

60

Total Marks in Writing: _____

This exam is a CLOSED book exam.
The exam comes in 10 pages (including cover page and 2 additional empty pages).

Question 1 [8 marks]

A. What is meant by criteria subsumption? Considering the different coverage criteria we studied in the course: mention a subsumption relationship where one coverage criteria subsumes another one, and **another** subsumption relationship where one coverage criteria does not subsume another one. You need to explain how the subsumption criteria is achieved or not achieved **[3]**.

B. Describe, and give examples to illustrate, the difference between a test requirement and a test case **[2]**.

C. Explain the difference between base choice coverage and multiple base choice coverage. Use an example to illustrate your explanation **[2]**.

D. Why would you need to apply the boundary value analysis technique, rather than relying only on the equivalence class partitioning technique? **[1]**

Question 2: Data Flow Coverage [15 Marks]

Consider the following source code. Answer **each** of the following questions:

```
/**
* Find index of pattern in subject string
*
* @param subject String to search
* @param pattern String to find
* return index (zero-based) of first
* occurrence of pattern in subject;
* -1 if not found
* @throws NullPointerException if subject
* of pattern is null
*/
public static int patternIndex (String subject, String pattern)
{
  final int NOTFOUND = -1;
  int iSub = 0, rtnIndex = NOTFOUND;
  boolean isPat = false;
  int subjectLen = subject.length;
  int patternLen = pattern.length;
  while (isPat == false && iSub + patternLen -1 < subjectLen)
  {
    if (subject.charAt (isSub) == pattern.charAt(0))
    {
      rtnIndex = iSub; // Starting at zero
      isPat = true;
      for (int iPat = 1; iPat < patternLen; iPat ++)
      {
        if (subject.charAt(iSub + iPat) != pattern.charAt(iPat))
        {
          rtnIndex = NOTFOUND;
          isPat = false;
          break; // out of for loop
        }
      }
    }
    iSub ++;
  }

  return (rtindex);
}
```

a) **(6 marks)** Draw the Control Flow Graph for the code. Use as minimal nodes as possible.

b) **(4 marks)** Decorate your CFG with Def-Use data for all variables.

c) **(5 marks)** Define all du-paths for the variables **iSub** and **Subject**.

Question 2: Input Space Partitioning [14 Marks]

a) **(2 marks)** A tester defined three characteristics based on the input parameter *car*: **Where Made**, **Energy Source**, and **Size**. The following partitionings for these characteristics have at least two mistakes. Correct them.

Where Made		
North America	Europe	Asia
Energy Source		
gas	electric	hybrid
Size		
2-door	4-door	hatch back

b) **(6 marks)** `NextDate` is a function that takes three arguments as input: month, day, year. It has the following specifications:

- It returns the date of the day <u>following</u> the input date. The allowed years are from 1812 – 2020.
- If it is not the last day of the month, only the day value will be incremented.
- At the end of a month, the next day is 1 and the month is incremented.
- At the end of the year, both the day and the month are reset to 1, and the year gets incremented.
- Leap year (سنة كبيسة) definition: A 29th day is added to February in all years that are evenly divisible by 4, except for centennial years (i.e., years that end with –00) which are not evenly divisible by 400. Hence, 1600, 2000 and 2400 are leap years, but 1700, 1800, 1900, 2100, 2200 and 2300 are not.

Identify the partitions for this function. **Note that you are not required to create concrete test cases.** You only need to generate the partitions.

c) (6 marks) A dialog box for modifying the font, allows making several changes, as follows:

- Font type: Can be either Calibri, Arial, or Times New Roman.
- Font size: Can be either large, or small.
- Font style: Can be either regular or italicized.
- Font highlighting: Can be either red or green.

i. What is the largest number of possible test cases needed to exhaustively test such a window? You need to explain your calculation that resulted in your answer.

ii. Apply pairwise coverage criteria to reduce the number of test cases as much as possible. You need to list all the tests that you will have after applying such a technique. You also need to show how you derived those tests.

Question 4: Structural Coverage [7 Marks]

Consider the given specifications for a (simplified) programmable thermostat. Suppose the variables that define the state and the methods that transition between states are:

```
partOfDay : {Wake, Sleep}
temp : {Low, High}
// Initially "Wake" at "Low" temperature
// Effects: Advance to next part of day
public void advance();
// Effects: Make current temp higher, if possible
public void up();
// Effects: Make current temp lower, if possible
```

i. Draw and label the states (with variable values) and transitions (with method names).

ii. Provide a test set that satisfies edge coverage on your graph.

Question 5: Logic Coverage [10 Marks]

a) **(2 MARKS)** Write the predicate to represent the requirement: "List all the wireless mice that either retail for more than $100 or for which the store has more than 20 items. Also list nonwireless mice that retail for more than $50."

b) **(8 marks)** Given that variables a,b are integers, consider the following expression:

```
((a == b) || (a < 5)) || ((a != b) && ( a + b == 10))
```

i. **(2 mark)** Explain the difference between clauses and the predicates within that expression.

ii. **(2 marks)** For the above expression, derive a set of test cases that achieves predicate coverage.

iii. **(1 mark)** What is meant by active clause coverage?

iv. **(3 marks)** Derive a set of test cases that achieves all Combinatorial Coverage.

Question 5: Mutation Testing [6 Marks]

Consider the following Java class listing that finds the index of the maximum value in an array:

```
1   public class FindMaxIndex {
2         static int indexer(int[ ]  a)    {
3               int i =0;
4               int index = 0;
5               while (i< a.length - 1)
6               {     i = i + 1;
7                     if (a [i] > a[index])
8                           index = i;
9               }
10              return index;
11        }
12  }
```

The following test suite (including three test cases) has already been developed by applying black-box testing techniques for this program:

T = {

Test case 1:<a[0] = 0, a[1] = 100, a[2] = 200>, expected behavior: Index of maximum value = 2,

Test case 2:<a[0] = 0, a[1] = 200, a[2] = 100>, expected behavior: Index of maximum value = 1,

Test case 3:<a[0] = 200, a[1] = 0, a[2] = 100>, expected behavior: Index of maximum value = 0,

}

Perform the mutation testing/analysis process on this program for the following three mutants. Note that each mutant is applied on the original program, and **NOT** on the previous mutants. Make sure you explain clearly in each stage if the mutant under analysis is killed or not. If the mutant is not killed, what test case should be added to kill it. Also, calculate the mutation score in each stage.

- Mutant 1: replace line 7 by: `if (a [i] >= a[index])`
- Mutant 2: replace line 4 by: `int index = a.length – 1;`
- Mutant 3: replace line 8 by: `index = i-1;`

End of Exam

Additional Space for the Student's use

Additional Space for the Student's use