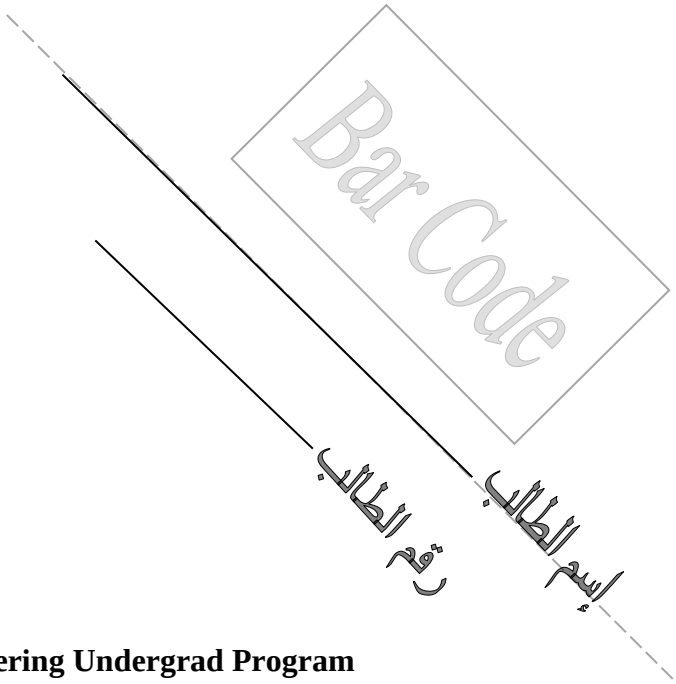


Cairo University
Faculty of Computers and Artificial Intelligence

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]**.

- Given criterion (c1)and criterion(c2) c1 subsume c2 if and only if every set of test cases satisfies c1 it also satisfies c2
in Graph coverage edge pair coverage subsume edge coverage because it will contain all sub paths of length 2 where edge coverage contains all sub paths of length 1
prime path don't subsume edge pair coverage because if we have sub path [n , n ,m] its satisfy edge pair coverage but it's not sample path or prime path (has internal loop)

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

Test Requirement : is specific element of software artifact that test case must be satisfy cover
Test case : is test related item which contains - set of test input (actual values) - execution condition - expected output

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

base choice : choice one base block from each characteristic and there is only one base test formed by using base choice for each characteristic
multiple base choice : at least one or more base choice blocks are chosen for each characteristic and one or more base tests formed by using base choices for each characteristic
MBCC subsume BCC Example in lecture

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

because
1 - instead of select an element from each block to be representative => this technique requires one or more elements to be selected so each edge of block be tested
2- instead of focusing on input conditions we also considering the result space (output)

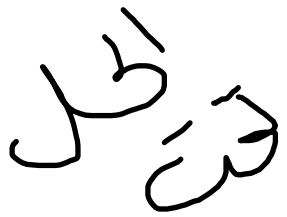
Question 2: Data Flow Coverage [15 Marks]

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

Pair Disjoint property violation: 2-door, 4-door blocks are not disjoint.
Size overlaps, a hatch-back could be 2-door or 4-door. Either add \2-door + hatch-back," and \4-door + hatch-back," or create two new characteristics:
Side Doors: 2, 4
Hatch-back: yes, no

Competence property violation: other made in countries are missing. Where Made is not complete. Add \other

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.

M1= {month: 1 <= month <= 12}
D1 = {day: 1 <=day<=31}
Y1= {year: 1812 <- year <- 2012}
And the invalid equivalence classes are:
M2 = {month < 1}
M3 = { month > 12}
D2 = {day < 1}
D3 = { day > 31}
Y2 = { year < 1812}
Y3 = { year > 2020}

Day:
D1: day between 1 to 28
D2: 29
D3: 30
D4: 31
Month:
M1: Month has 30 days
M2: Month has 31 days
M3: Month is February
Year:
Y1: Year is a leap year
Y2: Year is a normal year

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.

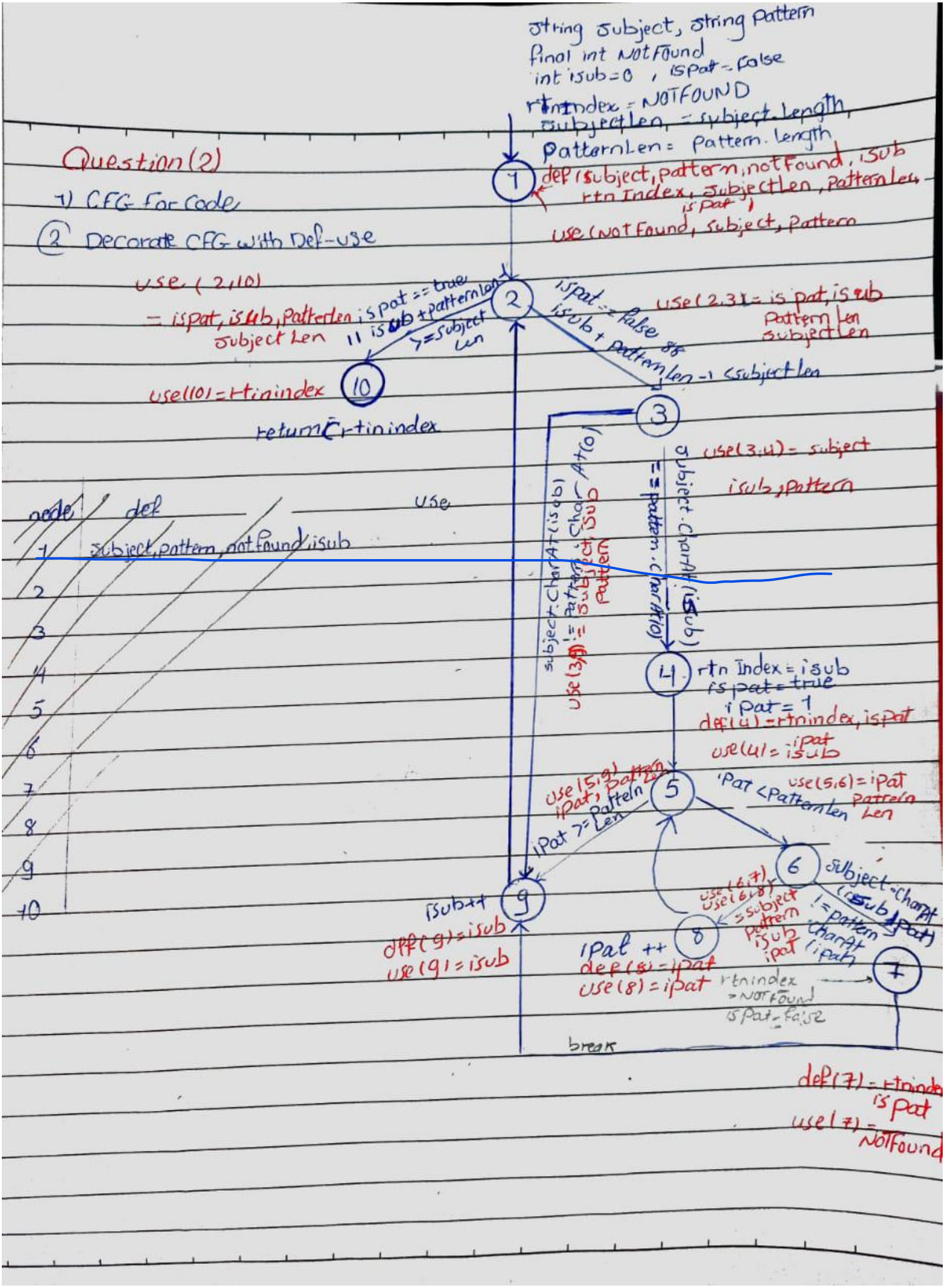
no. test cases = 3*2*2*2=24 test case

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 (2)

1) CFG For Code

2) Decorate CFG with Def-use



node	def	use
1	subject, pattern, notFound, isub	
2		
3		
4		
5		
6		
7		
8		
9		
10		

(3)

du pairs (iSub)	du path
(1, (2,3))	[1 2 3]
(1, (2,10))	[1 2 10]
(1, (3,4))	[1 2 3 4]
(1, (3,9))	[1 2 3 9]
(1, 4)	[1 2 3 4]
(1, (6,7))	[1 2 3 4 5 6 7]
(1, (6,8))	[1 2 3 4 5 8 8]
(1, 9)	[1 2 3 9]
(9, (2,3))	[9 2 3]
(9, (2,10))	[9 2 10]
(9, (3,4))	[9 2 3 4]
(9, (3,9))	[9 2 3 9]
(9, 4)	[9 2 3 4]
(9, (6,7))	[9 2 3 4 5 6 7]
(9, (6,8))	[9 2 3 4 5 6 8]

du pairs (subject)	du path need
(1, 1)	no path need
(1, (3,4))	[1 2 3 4]
(1, (3,9))	[1 2 3 9]
(1, (6,7))	[1 2 3 4 5 6 7]
(1, (6,8))	[1 2 3 4 5 6 8]

Font Type have 3 possible values [Calibri Arial Times new Roman]

Characteristic (x) $[x_1, x_2, x_3]$

Font size have 2 possible value [large, small]

Characteristic (y) $[y_1, y_2]$

Font style have 2 possible values [regular, italicized]

Characteristic (z) $[z_1, z_2]$

Font highlighting have 2 possible values [red, green]

Characteristic (δ) $[\delta_1, \delta_2]$

Pairs

x1 y1 — y1 z1 — z1 s1

x1 y2 — y1 z2 — z1 δ2

x1 z1 — y1 s1 — z2 δ1

x1 z2 — y1 δ2 — z2 δ2

x1 δ1 — y2 z1

x1 δ2 — y2 z2

x2 y1 — y2 s1

x2 y2 — y2 δ2

x2 z1

x2 z2 — Test Cases: — Cover

x2 δ1 — ① x1 y1 z1 s1 — (x1,y1) (x1 z1) (x1,s1) (y1,z1) (y1,s1) (z1 s1)

x2 δ2 — ② x1 y1 z1 δ2 — (x1,y1) (x1,z1) (x1,s1) (y1,z1) (y1,δ2) (z1,δ2)

x3 y1 — ③ x1 y1 z2 s1

x3 y2 — ④ x2 y2 z2 δ2

x3 z1 — ⑤ x2 y2 z1 s1

x3 z2 — ⑥ x2 y1

x3 s1 — ⑦ x3 y1 z1 s1

x3 δ2 — ⑧ x3 y2 z2 δ2

Additional Space for the Student's use

.

Additional Space for the Student's use