NATIONAL UNIVERSITY OF SINGAPORE

Department of Computer Science, School of Computing

IT5001—*Software Development Fundamentals*

Academic Year 2022/2023, Semester 1

Final Assessment

SOLUTIONS MANUAL

15 October 2022 Time Allowed: 2 hours

Instructions to Candidates (please read carefully):

1. This is a **closed-book assessment**. You are allowed **ONE (1)** A4-sized reference sheet, double-sided, printed or written, and **ONE (1)** blank A4 paper for scratch.
2. You may use a non-programmable calculator.
3. Use of any other electronic devices, including smart watches, is **NOT** allowed.
4. The assessment consists of three (3) documents—the 'QUESTION BOOKLET' (this document), the 'ANSWER BOOKLET', and an 'OCR Answer Sheet'. Do **NOT** open these documents until you are told to do so.
5. The 'QUESTION BOOKLET' (this document) comprises **FORTY-SIX (46)** questions and **EIGHT (8)** printed pages including this cover page.
6. The 'ANSWER BOOKLET' comprises **FOUR (4)** pages including the cover page.
7. Do **NOT** write your name on any document you submit.
8. Write your **Student Number (starting with A)** on the 'ANSWER BOOKLET'.
9. Write and shade your **Student Number (starting with A)** on the 'OCR Answer Sheet' **using a 2B pencil**.
10. For Multiple-Choice Questions (MCQs) Q1 to Q40 (inclusive), shade your answers on the 'OCR Answer Sheet' using a 2B pencil.
11. For the remaining questions, write your answers in the space provided in the 'ANSWER BOOKLET'; no extra sheets will be accepted as answers. You may write in blue or black with a pencil or pen.
12. You are required to only submit the 'ANSWER BOOKLET' and the 'OCR Answer Sheet' at the end of the assessment. You may use the 'QUESTION BOOKLET' (this document) as scratch paper.
13. The total attainable score for this assessment is **100 marks**. You must complete all questions to score full marks. This assessment counts towards **40%** of your final grade.
14. You **cannot** communicate with anyone other than the invigilators throughout the exam.
15. **You must attempt the assessment on your own.** The University takes a zero-tolerance approach towards plagiarism and cheating.

Section 1 MCQs (40 x 1 marks)

You can consider the code in each question is in a separate file and we run each from scratch. What will be the output when we run the file in IDLE? Write your answers in the **OCR form**.

	Questions	Answer
1	`print(9-+-+-+-9)`	a) 0 b) 9 c) -18 d) 18 e) Error
2	`print(4-5*2+3)`	a) -21 b) -9 c) -3 d) 1 e) None of the above
3	`print((True and False) or False)`	a) True b) False c) 1 d) 0 e) None of the above
4	`print(3>2 or 4<(1/0))`	a) True b) False c) 1 d) 0 e) None of the above
5	`print(bool(4)+int(True) or float(False))`	a) 0 b) 1 c) 1.0 d) 2 e) 5
6	`print(max('987','9111')+max('21','200'))`	a) 9111200 b) 9311 c) 98721 d) 1008 e) 2009111
7	`print('1234567890'[3])`	a) 0 b) 1 c) 2 d) 3 e) 4
8	`print('0'*1+2*'3')`	a) 06 b) 0003 c) 033 d) 0123 e) Error
9	`print('12345'[1:4][0:5][1][0])`	a) Empty string b) 2 c) 3 d) 4 e) Error
10	`print(['abc','def','ghi'][1][1])`	a) a b) b c) d d) e e) h

11	`print([9,[8,[7,6],[5,4]],3][1][2][0])`	a) Error b) 7 c) 6 d) 5 e) 4
12	`print(['123','456']['123'[0]])`	a) Empty String b) Error c) 456 d) 123 e) 4

13	`def f(x):`
	`  return x+3`
	`print(f(f(f(3))))`
		a) 6
		b) 9
		c) 12
		d) 15
		e) 18

14	`def f1(x):`
	`  return 1+f3(x)`
	`def f2(x):`
	`  return 2+f4(x)`
	`def f3(x):`
	`  return 1+f1(x)`
	`print(f1(1))`
		a) RecursionError
		b) NameError
		c) Infinite loop
		d) 8
		e) 6

15	`def f1(x):`
	`  return '1'+f2(x)`
	`def f2(x):`
	`  return f3(x)+'2'`
	`def f3(x):`
	`  return '3'+f4(x)`
	`def f4(x):`
	`  return '4'+x`
	`print(f1(0))`
		a) '13402'
		b) '3042'
		c) '13042'
		d) '12340'
		e) Error

16	`def f1(x):`
	`  if x > 0:`
	`  return f2(x-1)`
	`def f2(x):`
	`  if x > 0:`
	`  return f1(x-2)`
	`print(f1(6))`
		a) 0
		b) 2
		c) 6
		d) Error
		e) None

17	`def f(x):`
	`  if x <= 1:`
	`  return 1`
	`  if x%2:`
	`  return 1+f(x-3)`
	`  return 3+f(x-1)`
	`print(f(10))`
		a) 6
		b) 12
		c) 10
		d) 1
		e) 20

18	`x = 0`
	`for i in range(0,10,2):`
	`  x += 1`
	`print(x)`
		a) 0
		b) 2
		c) 5
		d) 9
		e) 10

19	`x = 0`
	`for i in range(3,11):`
	`  for j in range(6,10):`
	`  x += 1`
	`print(x)`
		a) 12
		b) 32
		c) 36
		d) 40
		e) 45

20	x,y = 0,0
	while x < 10:
	    x += 4
	    y += 1
	print(x)
		a) 2
		b) 3
		c) 4
		d) 8
		e) 12

21	x = 0
	for i in range(0,5,2):
	    while not i:
	        x += i
	print(x)
		a) 0
		b) 1
		c) 2
		d) 10
		e) Infinite loop

22	x = 10
	def foo(x):
	    x += 1
	    return x
	print(x+foo(x))
		a) 10
		b) 11
		c) 20
		d) 21
		e) Error

23	a = 3
	def foo(x):
	    a += 1
	    return x + a
	foo(a)
	print(a)
		a) 3
		b) 4
		c) 6
		d) 7
		e) Error

24	print(list({1:2,3:4}))	a) [1, 3] b) [(1,2),(3,4)] c) [2, 4] d) [{1:2,3:4}] e) Error

25	d = {0:2, 1:5, 2:1, 3:2, 4:1, 5:3, 3:9}
	a, output = 0, ''
	while a in d:
	    a = d[a]
	    output += str(a)
	print(output)
		a) '21539'
		b) '215639'
		c) '0215639'
		d) '2156347'
		e) Infinite loop

26	f1 = lambda x,y : x*y f2 = lambda a,b,c: a(b,c) print(f2(f1,3,4))	a) Error b) 9 c) 12 d) 16 e) a function
27	print(list(map(lambda x:x*2,'abc')))	a) ['abcabc'] b) ['aa', 'bb', 'cc'] c) ['abc', 'abc'] d) 'abcabc' e) Error
28	x = tuple(filter(lambda x:x%3,[2,4,6])) print(x)	a) (2, 4) b) (6,) c) (True, True, False) d) (2, 1, 0) e) (2, 4, 6)

29	class Pokemon:
	    def __init__(self,name,power):
	        self.name = name
	        self.slogan = name + ' ' + power
	Pikachu =
	Pokemon('Pikachu','electricity')
	Pikachu.name = 'Pika'
	print(Pikachu.slogan)
		a) Pikachu
		b) electricity
		c) Pikachu electricity
		d) Pika electricity
		e) Pika

30	class Animal:
	  def __init__(self):
	  self.sound = None
	  def speak(self):
	  print(self.sound)
	class Dog(Animal):
	  def __init__(self):
	  self.sound = "Woof"
	class Chihuahua(Dog):
	  def speak(self):
	  print("*cute*")
	  super().speak()
	dolly = Chihuahua()
	dolly.speak()
		a) None
		b) Woof
		c) *cute*
		d) *cute*
		  Woof
		e) Error

31	n=5
	try:
	  n+=1
	print(n+1)
	except:
	  print(n)
		a) 5
		b) 6
		c) 7
		d) Print nothing
		e) Syntax Error

32	n = 0
	try:
	  n += 1
	except:
	  n += 2
	else:
	  n += 3
	finally:
	  n += 4
	print(n)
		a) 1
		b) 5
		c) 7
		d) 8
		e) 10

33	try:
	  x = 1
	  assert False
	except AssertionError:
	  x += 2
	except:
	  x += 4
	print(x)
		a) 1
		b) 3
		c) 5
		d) 7
		e) Crash and print nothing

34	The two file opening modes, 'w+' and 'r+', have totally same functionalities.	a) True b) False
35	What is/are the difference(s) of the two file opening modes 'r+' and 'a+'? 1. If the file didn't exist, 'r+' will raise an error and 'a+' will not 2. The initial file pointer location 3. 'r+' can read the file and 'a+' cannot	a) 1 only b) 2 only c) 1 and 2 only d) 2 and 3 only e) 1, 2 and 3
36	What is the best and worst time complexities for MergeSort respectively with array size n?	a) O(1) and O(n) b) Both O(n) c) O(n) and O(n log n) d) Both O(n log n) e) O(n log n) and $O(n^2)$

37	In our lecture, what is the name of the technique to speed up Fibonacci number computation with storing pre-computed answers in dictionary?	a) Caching b) Dictionary lookup c) Memoization d) Dynamic Programming e) Divide-and-conquer
38	If we want to write a function for (calculus) integration approximation, which technique is the most suitable?	a) Bisection method b) Polymorphism c) Higher Order Function d) Inheritance e) Function Abstraction
39	L1 = [1,2] L2 = [L1,3,4] L3 = L2.copy() L3[0][1] = 5 print(L2)	a) [5, 3, 4] b) [[1, 2], 3, 4] c) [[5, 2], 3, 4] d) [[1, 2], 5, 4] e) [[1, 5], 3, 4]
40	L1 = ([1,2],3,4) def f(L): L[0][1] = 9 f(L1) print(L1)	a) Error b) ([1, 2], 3, 4) c) ([1, 9], 3, 4) d) ([9, 2], 3, 4) e) [[9, 2], 3, 4]

Section 2 Long Questions (60 marks)

Please write your answers in the **answer booklet**.

Question 41 (10 marks)

Here is the same partial code from your OOP assignments.

```
class Character(object):
    def __init__(self):
        self.name = ''
        self.maxhp = 1000
        self.hp = 1000
        self.str = 0
        self.maxmana = 0
        self.mana = 0
        self.cost = 9999999999
        self.alive = True

    def act(self,myTeam,enemy):
        return

    def gotHurt(self,damage):
        if damage >= self.hp:
            self.hp = 0
            self.alive = False
            dprint('I died!')
        else:
            self.hp -= damage
            dprint(f'hp:{self.hp}.')
```

```
class Fighter(Character):
    def __init__(self):
        super().__init__()
        self.name = 'Fighter'
        self.maxhp = 1200
        self.hp = 1200
        self.str = 100
        self.cost = 100

    def act(self,myTeam,enemy):
        target = randAlive(enemy)
        dprint(f'Damage {self.str}.')
        enemy[target].gotHurt(self.str)
```

Write a new class Ninja. He can do what a fighter do. The difference is

- His hp is only 600 but his cost is 150
- He has a 50% chance to dodge his enemy attack and he is not hurt at all
- When he attacks, he has a 10% chance that he can do a double attack. Namely, he can attack two enemies or attack the same enemy twice.

Write your new class Ninja in the answer booklet in the best OOP practice and style. You can assume the random package is imported.

Question 42 (10 marks)

Oh no, the code on the right does not work! The function `flip()` is supposed to reverse the input list and return it, but when we try with `flip(['a','b','c'])`, the returned result is still `['a','b','c']`

```
def swap(lst, i, j):
  lst[i],lst[j] = lst[j],lst[i]
  return lst
def flip(lst):
  n = len(lst)
  for i in range(n):
    n -= 1
    swap(lst,i,n)
  return lst
```

Point out and describe the problem(s) of this code. Why is it not working as expected? Suggest a change to the code that is minimal to make it work. (Definitely not "`return lst[::-1]`"!). You do not need to rewrite the code, describing your changes will be good enough.

Question 43 (10 marks)

The following code runs perfectly for computing the balance after one year with the given starting principal and interest rate. However, the code is not really in a good style. Rewrite the following code with our good abstraction principles. You can assume the user always input floating point numbers.

```
#compute the balance after 1 year
while True:
  principal = float(input("Enter the starting principal:"))
  if principal > 0:
    break
while True:
  interest = float(input("Enter the interest rate:  "))
  if interest > 0:
    break
print("After one year")
print("You will have", principal*(1+interest))
```

Question 44 (10 marks)

Given a sequence of integers, Write the function `three_sum(L,target)` to return a list of all triplets (3 numbers) in the input that sum up to a given target value. You can assume the input integers are all unique, namely, no two integers are the same. Your function should have some sample output like this.

```
>>> three_sum([4,3,1,2,7,8,5,9],10)
[(4, 1, 5), (3, 2, 5), (1, 2, 7)]
>>> three_sum([4,3,1,2,7,8,5,9],18)
[(4, 5, 9), (3, 7, 8), (1, 8, 9), (2, 7, 9)]
```

Question 45 (10 marks)

You are given a DNA sequence `seq`, given a subsequence `subseq`, write a function `seq_count(seq,subseq)` to return the number of `subseq` occurrence in `seq`. Note that you can have two occurrences, can share some common parts.

```
>>> seq_count('ACTACTAG','ACTA')
2
>>> seq_count('AAAAAAA','AAAA')
4
```

Question 46 (10 marks)

What is the output for the following code?

```
class Speaker:
  def say(self, stuff):
    print(stuff)
class RepeatingLecturer(Speaker):
  def say(self, stuff):
    super().say(stuff)
    print("I repeat again")
    super().say(stuff)
class BabySpeaker(Speaker):
  def say(self, stuff):
    print("Baby shark dodododododo...")
    super().say(stuff)
class RepeatingBabySpeaker(RepeatingLecturer, BabySpeaker):
  pass

alan = RepeatingBabySpeaker()
alan.say("OOP is fun!")
```

-- End of Paper --

Solutions

1. D	2. C	3. B	4. A	5. D	6. C	7. E	8. C
9. C	10. D	11. D	12. B	13. C	14. A	15. E	16. E
17. B	18. C	19. B	20. E	21. E	22. D	23. E	24. A
25. A	26. C	27. B	28. A	29. C	30. D	31. E	32. D
33. B	34. B	35. C	36. D	37. C	38. C	39. E	40. C

Question 41 (Hint: Please use the grey lines for your indentation alignments)

```
class Ninja(Fighter):
    def __init__(self):
        super().__init__()
        self.name = 'Ninja'
        self.hp = 600
        self.maxhp = 600
        self.cost = 150
    def act(self, myTeam, enemy):
        if random.randint(1,10)==1:
            super().act(myTeam, enemy)
        super().act(myTeam, enemy)
    def gotHurt(self, damage):
        if random.randint(1,2)==1:
            super().gotHurt(damage)
        else:
            dprint('Dodged!')
```

Question 42

The problem lies in the line:

```
for i in range(n):
```

That makes the code to reverse the list twice

Just change it to

```
for i in range(n//2):
```

Question 43

```
def request_number(prompt):
  while True:
    number = float(input(prompt))
    if number > 0:
      break
principal = request_number("Enter the starting principal:")
interest = request_number("Enter the interest rate:  ")
print("After one year")
print("You will have", principal*(1+interest))
```

Question 44

```
def three_sum(L,target):
  output = []
  for i in range(len(L)-2):
    for j in range(i+1,len(L)-1):
      for k in range(j+1,len(L)):
        if L[i]+L[j]+L[k] == target:
          output.append((L[i],L[j],L[k]))
  return output
```

Question 45

```
def seq_count(seq,subseq):
  lss = len(subseq)
  count = 0
  for i in range(len(seq)):
    if subseq == seq[i:i+lss]:
      count += 1
  return count
```

Question 46

```
Baby shark dododododo...
OOP is fun!
I repeat again
Baby shark dododododo...
OOP is fun!
```