NATIONAL UNIVERSITY OF SINGAPORE

Semester 1, 2019/2020

CS1010E – PROGRAMMING METHODOLOGY

Time Allowed: 2 Hours

INSTRUCTION TO CANDIDATES

1. This assessment paper contains **TWENTY-FOUR (24)** questions and comprises **THIRTEEN (13)** printed pages, including this page. The maximum possible score is 100 marks.
2. This is a **CLOSED BOOK** (WITH AUTHORIZED MATERIALS) assessment. You are allowed to bring in one piece of A4 size reference sheet.
3. Calculators are allowed, but not programmable ones, laptops or other electronic devices.
4. Answer all **MCQ questions** by shading the letter corresponding to the most appropriate answer on the OCR form provided. Shade and write down your **student number** on the **OCR form** as well. Use a 2B pencil to shade on the OCR form, or the grading machine might not be able to register your shading.
5. Answer all programming question **on this question paper**. Write down your student number at this page, but do not write your name.
6. We will **collect both** the OCR form and question paper at the end of the assessment.
7. Leave your student card on the desk throughout this assessment.
8. In the programming question, the light grey lines are guidelines for indentations
9. You can use the last blank page of the question paper to answer. However, please add appropriate "pointers" from the question, e.g. "*Please refer to my answer at…*"
10. You are allowed to use pencils, but please use pens as much as possible.
11. And we will penalise for code that is redundant, hard-coded or too lengthy.
12. Do NOT look at the questions until you are told to do so.

STUDENT NUMBER:_____

(This portion is for examiner's use only)

Question	Max. Marks	Score	Check
Q1-20	60		
Q21	10		
Q22	10		
Q23	10		
Q24	10		
Total	100		

Part I. Multiple Choice Questions (MCQs) [Total: 3 x 20 = 60 marks]

Unless explicitly mentioned, each MCQ has one correct answer. Every question is worth 3 mark. There is no penalty for wrong answers, and there is no mark given for partially correct answers. Answer all the MCQ question in the OCR form provided.

1. What value is produced from executing the following expression?

```
2 ** (1 if ('0' if 'False' else '1') else 0) ** 3
```

(A) 1	**(B) 2**
(C) 8	**(D)** Run-time error
(E) None of the above	

2. Given the following two assignments to variable x, what integer value should x have **before** assignment in order for the same value to be assigned to x after executing each of these assignments?

```
x *= x + 2                    x = x * x + 2
```

(A) 0	**(B) 1**
(C) All possible integer values	**(D)** More than one possible answer in choice **(A), (B), (C)**
(E) None of the above	

3. Given the following function definition of **`shift`**, which of the following applications of **`shift`** will be evaluated successfully?

```
def shift(t,word):
    return t[2:] + (word,)
```

(A) `shift('abc','d')`	**(B)** `shift(['w1', 'w2'], 'w3')`
(C) `shift(('1',),[[3]])`	**(D)** `shift( (1), (2) )`
(E) None of the above	

4. Given the following Ackermann function definition **`ack`**. During the execution of the function application **`ack(1,2)`**, how many time is the function **`ack`** being call (including the first call **`ack(1,2)`**)?

```
def ack(m,n):
    if m == 0:
        return n+1
    elif m > 0 and n == 0:
        return ack(m-1,1)
    else:
        return ack(m-1,ack(m,n-1))
```

(A) 4	**(B) 6**
(C) 8	**(D)** Infinite, since execution enters infinite loop
(E) None of the above	

5. The following function definition has the following input-output behavior:

```
>>> lst = [-25, -10, -7, 2, 23, 4, -5, 10]
>>> madness(lst)
[(-25, 2, 23)]
```

```
def madness(nums):
    nums = sorted(nums)
    res = {}
    i = 0
    while i < len(nums)-2:
        j, k = i+1, len(nums)-1
        while j < k:
            if nums[i] + nums[j] + nums[k] == 0:
                res[(nums[i],nums[j],nums[k])] = True
                j += 1
                k -= 1
            elif (nums[i] + nums[j] + nums[k] < 0):
                j += 1
            else:
                k -= 1
        i += 1
    return list(res.keys())
```

What does the function do?

A. It returns a tuple containing the minimum, maximum, and middle elements in the list

B. It returns just one tuple of three elements in the input list the sum of which is 0.

C. It returns all tuples of three elements in the input list the sum of which is 0.

D. It returns some, but may not necessarily all, tuples of 3 elements in the input list the sum of which is 0.

E. None of the above

6. Consider the following function definition, what is the result of evaluating the following expression?

```
(lambda x: (lambda x : x)(lambda x: x+x)(x))(2)
```

(A) 0	**(B) 2**
(C) 4	**(D) (lambda x:x+x)**
(E) Run time error occurs	

7. Consider the following function definition, what is the result of execution the function call `crack(8, lambda x: x)`??

```
def crack(n,g):
    if n % 5:
        return crack(n+1,g(g))
    else:
        return g(n)
```

(A) 10	**(B) 9**
(C) 8	**(D)** Run-time error
(E) None of the above	

8. Given that **oldmat** is a 4x4 integer matrix in which every integer in the matrix is unique. After executing the following code fragment, where is the value of **oldmat[0][0]** stored in the resulting matrix, **newMat**?

```
def confuse(mat):
    return [ mat[len(mat)-i-1][::-1]
             for i in range(len(mat)) ]

newMat = confuse(oldmat)
```

(A)	**newMat[0][0]**	**(B)**	**newMat[0][3]**
(C)	**newMat[3][0]**	**(D)**	**newMat[3][3]**
(E)	Run time error		

9. Consider the following class definitions.

```
class NUS:
    def id(self):
        return 'NUS'
class SoC(NUS):
    def id(self):
        return 'SoC'
class FOE(NUS):
    def id(self):
        return 'FOE'
class Student(FOE,SoC):
    pass
```

What will be printed from executing the following statements:

```
>>> peter = Student()
>>> print(peter.id())
```

(A)	**NUS**	**(B)**	**SoC**
(C)	**FOE**	**(D)**	**FOESoCNUS**
(E)	**FOENUS**		

10. The following definition of function **memof** makes use of a dictionary **memo** in its computation. After evaluating the call **memof(5)**, what will be the dictionary look like (ignoring the ordering of the items in the dictionary)?

```
memo = {0:0, 1:1, 2:2}
def memof(n):
    if n in memo:
        return memo[n]
    res = memof(n-1) + memof(n-3)
    memo[n] = res
    return res
```

(A)	{0:0,1:1,2:2}	**(B)**	{0:0,1:1,2:2,3:2,4:3,5:5}
(C)	{0:0,1:1,2:2,3:3,4:5,5:8}	**(D)**	{0:0,1:1,2:2,3:2,4:3}
(E)	{0:0,1:1,2:2,3:3,4:5,5:7}		

11. Consider the following definition of **updLst**, and the subsequent assignment statements which invoke **updLst** function:

```
def updLst(lst):
    nlst = []
    for i in range(len(lst)-1,-1,-1):
        if type(lst[i]) == list:
            nlst.append(lst[i])
            for j in range(len(lst[i])):
                lst[i].pop()
        else:
            nlst.append(lst[i])
    return nlst

lst1 = [1,[2,3],[4,[5],6],7]
lst2 = updLst(lst1)
```

What will be the content of **lst1** and **lst2**, after executing the last two assignment statements?

	lst1	Lst2
A.	`[1,[2,3],[4,[5],6],7]`	`[7, [], [], 1]`
B.	`[1, [], [], 7]`	`[7, [4,[5],6], [2,3], 1]`
C.	`[1,[2,3],[4,[5],6],7]`	`[7, [4,[5],6], [2,3], 1]`
D.	`[1, [], [], 7]`	`[7, [], [], 1]`
E.	None of the above	

12. What is the output from running the following code fragment?

```
class P():
    def function1(self):
        print('class P')
class Q(P):
    def function1(self):
        super().function1()
        print('class Q')
class R(Q):
    def function1(self):
        print('class R')
        super().function1()
r = R()
r.function1()
```

A.	`class R`
	`class P`
	`class Q`
		B.	`class R`
			`class Q`
			`class P`

C.	`class P`
	`class R`
	`class Q`
		D.	`class Q`
			`class P`
			`class R`

E.	None of the above		

13. Consider the following function definitions for **myzip** and **distance**. Which of the following calls to distance will evaluate to the number **2**.

```
def myzip(w1,w2):
    def myzip1(i,m,n):
        if i == m or i == n:
            return []
        return [(w1[i],w2[i])]+myzip1(i+1,m,n)
    return myzip1(0,len(w1),len(w2))
def distance(w1,w2):
    cnt = 0
    for c1,c2 in myzip(w1,w2):
        if c1 != c2:
            cnt += 1
    return cnt
```

(A) `distance("anagram","anamarg")`	**(B)** `distance("anagram","margana")`
(C) `distance("anagram","anagrames")`	**(D)** `distance("anagram","anagarmmes")`
(E) None of the above	

14. The following code fragment attempts to create a matrix of **n** rows. What will be printed as the content of **mat[1]**?

```
def makeMat(n):
    mat = [[]] * n
    count = 0
    for row in mat:
        for _ in range(2):
            row.append(count)
            count += 1
    return mat

mat = makeMat(2)
mat[1]
```

(A) **[ ]**	**(B)** **[2, 3]**
(C) **[0, 1, 2, 3]**	**(D)** Run time error
(E) None of the above	

15. What is the value returned by evaluating the following expression?

```
list(filter (lambda x : 10 % x,
             filter (lambda x: x % 2,[1,2,3,5,7,64])))
```

(A) **[0, 1, 0, 3]**	**(B)** **[64]**
(C) **[3, 7]**	**(D)** Zero division error occurs
(E) None of the above	

16. The following code fragment defines a function **scan** and an addition **add**, such that

scan(add,0,[1,2,3,4,5]) produces **[1,3,6,10,15]**

scan(add,0,[1,-2,3,-4,5]) produces **[1,-1,2,-2,3]**

That is, element in the resulting list indexed at i is produced by summing up elements indexed by $0,1,\ldots,i$ in the third argument (a list) of **scan**.

```
def scan(f,acc,lst):
    def scan1(f,acc,i,n):
        if i >= n:
            return []
        item = f(lst[i],acc)
        return XXXXX
    return scan1(f,acc,0,len(lst))
add = lambda x,y:x+y
```

Complete the definition of **scan** by replacing **XXXXX** in its definition the most appropriate expression below.

A. `[item].extend(scan1(f,item,i+1,n))`

B. `[item]+scan1(f,acc,i+1,n)`

C. `[acc]+scan1(f,item,i+1,n)`

D. `scan1(f,item+acc,i+1,n)`

E. None of the above

17. Which of the following statements is correct after running the code fragment below?

```
class Par:
    def __init__(self):
        self.calc(30)
    def calc(self, i):
        self.aa = 2 * i;
class Chi(Par):
    def __init__(self):
        super().__init__()

    def calc(self, i):
        self.bb = 3 * i;

kid = Chi()
```

A. **kid.aa** has value 60

B. **kid.bb** has value 90

C. **kid.aa** and **kid.bb** have values 60 and 90 respectively

D. Run-time error occurs

E. None of the above

18. Consider the following code fragment, given that **`lst = [3,5,7,9]`**, what is the result produced by executing **`squeeze(lst,2,1)`**?

```
def squeeze(lst,n,a):
    done = False
    while n >= 0 and not done :
        if lst[n] > a:
            lst[n+1] = lst[n]
            n -= 1
        else:
            lst[n+1] = a
            done = not done
    if not done:
        lst[0] = a
    return lst
```

(A)	`[3, 5, 7, 1]`	**(B)**	`[1, 3, 5, 7]`
(C)	`[1, 3, 5, 7, 9]`	**(D)**	Run-time error was reported
(E)	None of the above		

19. Given the following code fragment:

```
class P():
    def __init__(self):
        self.x = 0
class Q():
    def __init__(self):
        self.y = 0

class R(P,Q):
    def get_x(self):
        return self.x
r = R()
```

Which of the following expressions will be evaluated to True?

(A)	`isinstance(r,Q)`	**(B)**	`subclass(R,P)`
(C)	`inherited(R,Q)`	**(D)**	More than one of the choices A, B and C above can be evaluated to **`True`**
(E)	None of the above		

20. Given the following function definition.

```
def combinator(y):
    return (lambda x: lambda y: x(y))(lambda x:y)
```

What is the result of evaluating the expression

`combinator(lambda x:x*10)(11)(12)`?

(A)	`100`	**(B)**	`120`
(C)	`132`	**(D)**	Run time error is encountered
(E)	None of the above		

Q1 – Q20 Answers:

Q1	**B**
Q2	**B**
Q3	**C**
Q4	**B**
Q5	**C**
Q6	**C**
Q7	**A**
Q8	**D**
Q9	**C**
Q10	**B**
Q11	**D**
Q12	**A**
Q13	**D**
Q14	**C**
Q15	**C**
Q16	**E**
Q17	**B**
Q18	**B**
Q19	**A**
Q20	**B**

Part II. Programming Question [Total: 40 marks]

Write down the program on the answer sheet. Partial mark may be awarded to your program.

Question 21 Deep Reverse (10 marks)

Implement a function `deepReverse` such that it will reverse a list of lists in a deep manner like all the deep functions such as `deepMap`. e.g.

```
>>> l = [1,2,[3,4,[5,6,[7],8],9],10,[11]]
>>> print(deepReverse(l))
[[11], 10, [9, [8, [7], 6, 5], 4, 3], 2, 1]
>>> print(deepReverse(deepReverse(l)))
[1, 2, [3, 4, [5, 6, [7], 8], 9], 10, [11]]
```

You must following the coding style below and fill in the empty spaces without changing the existing code or importing any packages/functions. You can assume that the input has no tuples in it.

```
def deepReverse(seq):
    if seq == []:
        return seq

    elif type(seq) != list:
        return seq

    else:
        return deepReverse(seq[1:]) + [deepReverse(seq[0])]
```

Question 22 Local Peak (10 marks)

Given a list of numbers `lst`, we call a number `lst[i]` is a local peak if all of its neighbor(s) are smaller than it. If `i = 0` or `len(lst)-1`, we only have to consider one neighbor. We can assume the list contains at least two numbers and no two consecutive numbers have the same value. The expected output should be the index of the *first* local peak as follows.

```
>>> print(localPeak([1,5,6,9,5,4,3,10,11]))
3
>>> print(localPeak([1,2,3,4,5]))
4
>>> print(localPeak([10,9,8,7]))
0
```

However, Mr Silly wrote the following function:

Line#	Code:
1	`def localPeak(lst):`
2	`    l = len(lst)`
3	`    if lst[0] > lst[1]:`
4	`        return 0`
5	`    if lst[l-1] > lst[l-2]:`
6	`        return l-1`
7	`    for i in range(1,l-1):`
8	`        if lst[i-1] < lst[i] and lst[i] > lst[i+1]:`
9	`            return i`

Question 22 (Cont.)

Assuming all the indentations and syntax are correct, and the code can run without crashing. Also assume that the input list has more than one element. Name one mistake that Mr. Silly made.

For the first example output, the function by Mr Silly will return the last local Peak instead of the first one.

And please suggest how to correct the above mistake and where this correction should be done in the code. You can draw some doodle at the above code to help your illustration.

Move Lines 5 and 6 to the end of the code

Question 23 (5 + 5 marks)

The hyperbolic sine function is given as followings

$$\sinh x = \sum_{n=0}^{\infty} \frac{x^{2n+1}}{(2n+1)!}$$

As usual, we will not sum all the terms from 0 to infinity, but we will sum 50 terms only.
Write an iterative version of `sinhI(x)` to compute the above function by summing up 50 terms only (n from 0 to 49) without using recursion. Your function must have some (meaningful) loop(s) in it.

```
from math import factorial

def sinhI(x):
    ans = 0
    for i in range(50):
        ans += x**(2*i+1)/factorial(2*i+1)
    return ans
```

Write an "ultimate" version of `sinhU(x)` to compute the above function by summing up 50 terms only (n from 0 to 49) WITHOUT using any iteration (for-loop or while-loop) or recursion. Hint: You can use "`for`" in list comprehensions.

```
from math import factorial

def sinhU(x):
    def sf(i):
        return x**(2*i+1)/factorial(2*i+1)
    return sum(map(sf,range(0,50)))
```

Question 24 (5 + 5)

Remember our RPG assignment? We are going to build more characters on top of that in this question. You can assume the function `randint(a,b)` (that generates a random integers x for a <= x <= b) is imported from the package `random`. ***Your code should demonstrate good OOP practices.***

Question 24 Task 1

Write the class for a new type of character `ClumsyFighter`. He will have the same parameters (str, wis, hp and so on) as a `Fighter`, except that the class name is '`ClumsyFighter`' and his cost is only 50. When he acts, he will have a 25% chance of tripping himself and got himself hurt instead of attacking enermies. When this happens, he will have a damage of 50 to *himself*. Otherwise, there is a 75% chance that he will attack like what a normal Fighter does.

```
class ClumsyFighter(Fighter):
    def __init__(self):
        super().__init__()
        self.name = 'ClumsyFighter'
        self.cost = 50
    def act(self,myTeam,enemy):
        if randint(0,3):
            dprint("Oops! Hurt myself!")
            self.gotHurt(50)
        else:
            super().act(enemy,myTeam)
```

Question 24 Task 2

Write the class for a new type of character **`Cleric`**. He will have the same parameters (str, wis, hp and so on) as a Mage, except that the class name is '`Cleric`' and his cost is 300. The only difference is, his spell will not damage the enemies. Instead, he will add 300 hp to a random member in his own team who is alive (including himself). If anyone's hp exceed maxhp after healing, his hp will be set to maxhp. The healing will cost mana like the spell by the class Mage, and obey the same rule if his mana is not enough.

```
class Cleric(Mage):
    def __init__(self):
        super().__init__()
        self.name = 'Cleric'
        self.cost = 300
    def cast(self,myTeam,enemy):
        self.mana -= manaCost
        target = randAlive(myTeam)
        myTeam[target].hp +=100
        dprint("Healing!")
        if myTeam[target].hp > myTeam[target].maxhp:
            myTeam[target].hp = myTeam[target].maxhp
```

This is the skeleton code given in the assignment for your reference.

```
# The two variables ‘myTeam’ and ‘enemy’ should be a list
# of instances of class Character of its subclasses

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(self.name + ' died!')
        else:
            self.hp -= damage
            dprint(self.name + f' hurt with remaining hp {self.hp}.')
class Fighter(Character):
    def __init__(self):
        super().__init__()
        self.name = 'Fighter'
        self.hp = 1200
        self.maxhp = 1200
        self.str = 100
        self.cost = 100

    def act(self,myTeam,enemy):
        target = randAlive(enemy)
        dprint(f'Hurt enemy {target} by damage {self.str}.')
        enemy[target].gotHurt(self.str)
class Mage(Character):
    def __init__(self):
        super().__init__()
        self.name = 'Mage'
        self.maxmana = 50
        self.mana = 50
        self.hp = 800
        self.maxhp = 800
        self.cost = 200
        self.int = 400

    def cast(self,myTeam,enemy):
        self.mana -= manaCost
        target = randAlive(enemy)
        dprint(f'Strike enemy {target} with spell')
        enemy[target].gotHurt(self.int)

    def act(self,myTeam,enemy):
        if self.mana < manaCost:
            self.mana += manaRecovery
            dprint(f'Mana recover to {self.mana}.')
        else:
            self.cast(myTeam,enemy)
```

=== END OF PAPER ===