Student ID : _____

CITY UNIVERSITY OF HONG KONG

Course code & title : CS1302 Introduction to Computer Programming

Session : Semester B 2021-22

Time allowed : Two (2) Hours

This paper has **14** pages (including this page).

Note:

1. Answer ALL the questions in the space provided within each question.

Question	Part 1 (10 marks)	Part 2 (30 marks)	**Total (40)**
Marks			

2. Do NOT remove the staple or separate the paper.
3. This question paper should NOT be taken away.

Remarks:

For all written code required by the questions:

1. You should give precise Python code with proper programming styles, in particular, appropriate code design, naming and code formatting. Marks may be deducted for redundant or unnecessary code.
2. Unless specifically mentioned, modules/libraries/functions other than built-in functions, math modules are not allowed.
3. Please write down your answer on your A4 or single line paper.

Note that this is just a sample exam paper. It doesn't represent all the question types/numbers/difficulty of the real final exam.

---

*This is a **closed-book** examination.*

No materials or aids are allowed during the whole examination. If any unauthorized materials or aids are found on a candidate during the examination, the candidate will be subject to disciplinary action.

Part 1 Multiple Choice Questions (10 marks)

Multiple choice questions (1 mark for each question).

Note that there may be more than 1 correct option for each question, and you must select all the correct options to get the full mark. There is no partial mark such as 0.5. The text in brown color is Python code.

(1) What data type is the object below? L = [1, 23, 'hello', 1]?

A. List
B. Dictionary
C. Tuple
D. Array

Correct answer: A

(2) Which of the following function convert a string to a float in python?

A. int(x [,base])
B. long(x [,base])
C. float(x)
D. str(x)

Correct answer: C

(3) What is the output of the following segment: chr(ord('A'))

A. A
B. B
C. a
D. Error

Correct answer: A

(4) Which of the following is False regarding loops in Python?

A. Loops are used to perform certain tasks repeatedly.
B. While loop is used when multiple statements are to executed repeatedly until the given condition becomes False
C. While loop is used when multiple statements are to executed repeatedly until the given condition becomes True.
D. for loop can't be used to iterate through the elements of lists.

Correct answer: B, D

(5) Among the following symbols, the one-line comment in Python is NOT ().

A. #
B. //
C. <!-->
D. ””

Correct answer: B C D

(6) Suppose list1 is [3, 4, 5, 20, 5, 25, 1, 3], what is list1 after list1.pop(1)?

A. [3, 4, 5, 20, 5, 25, 1, 3]
B. [1, 3, 3, 4, 5, 5, 20, 25]
C. [3, 5, 20, 5, 25, 1, 3]
D. [1, 3, 4, 5, 20, 5, 25]

Correct answer: C

(7) Suppose we have already excuted the following code. Which of the following statements are correct?.

```
a = [x**2 for x in range(1,4)]

b = a[::-1]

c = reversed(a)
```

A. Continue to execute print(b) gives [9 4 1].
B. Continue to execute print(c) gives [9 4 1].
C. c is an iterable object.
D. Continue to excute print(d) gives [9 4 1].

Correct answer: A, C

(8) Which statement below is correct?

A. Both `list` and `tuple` objects may be changed.
B. `tuple` has the method `append` to append new element.
C. A mutable item of a tuple can be modified but we cannot change the tuple to point to a different item.
D. A tuple can be inserted into a list.

Correct answer: C, D

(9) What is the output of the following code?

```
i = 0
while i < 3:
    print(i)
    i = i+1
    print(i+1)
```

A. 0 2 1 3 2 4
B. 0 1 2 3 4 5
C. Error
D. 1 0 2 4 3 5

Correct answer: A

(10) What is the output of the following code?

```
def myfunc(a):
  a = a + 2
    a = a * 2
```

A. 8
B. 16
C. Indentation Error
D. Runtime Error

Correct answer: C

Part 2 Coding (30 marks)

Please complete the following programming questions (3 marks for each question, in total 10 coding questions)

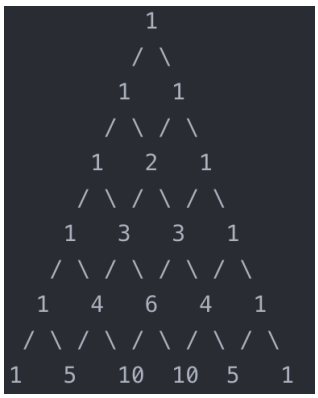
1. Define a function triangles() to show the Pascal's Triangle.

Given a non-negative integer numRows, generate the first numRows rows of "Pascal's Triangle".

In the Pascal's Triangle, each number is the sum of its upper left and upper right numbers.

The n layer (the top layer is called layer 0, line 1, the n layer is n+1 line, where n is a natural number including 0) corresponds exactly to the coefficient of the binomial $(a+b)^n$ expansion. For example, the second level 1 2 1 is the expanded form of the binomial $(a+b)^2$ with a power of 2 The coefficient of $a^2+2ab+b^2$.

Treat each line as a list, try to write a generator, and continuously output the list of the next line



For example,

Test	Result
x = triangles() print(next(x)) print(next(x)) print(next(x)) print(next(x))	 [1] [1, 1] [1, 2, 1] [1, 3, 3, 1]

Answer:

```
def triangles():
    L = [1]
    while True:
        yield L
        L.append(0)
        L = [L[i - 1] + L[i] for i in range(len(L))]
```

2. Define a function max_sublist_sum that accepts an argument that keeps a sequence of numbers and returns the maximum sum of the nonempty contiguous subsequence.

   For example, when [-6, -4, 4, 1, -2, 2] is given as the argument, the function returns 5 because the nonempty subsequence [4, 1] has the maximum sum 5.

   For example:

Test	Result
print(max_sublist_sum([-6, -4, 4, 1, -2, 2]))	5
print(max_sublist_sum([2.5, 1.4, -2.5, 1.4, 1.5, 1.6]))	5.9

Answer:

```
def max_sublist_sum(seq):
    return max(sum(seq[i:j]) for i in range(len(seq)) for j in range(i+1,len(seq)+1))
```

3. Define a function called merge that

- takes two sequences sorted in ascending orders, and
- returns a sorted list of items from the two sequences.

Then, define a function called mergesort that

- takes a sequence, and
- return a list of items from the sequence sorted in ascending order.

The list should be constructed by

- recursive calls to mergesort the first and second halves of the sequence individually, and
- merge the sorted halves

For this question, **do not use the sort method or sorted function.*

For example:

Test	Result
assert merge([1,3],[2,4]) == [1,2,3,4]	
assert mergesort([3,2,1]) == [1,2,3]	
assert mergesort([3,5,2,4,2,1]) == [1,2,2,3,4,5]	

Answer:

```
def merge(left,right):
    if left and right:
        if left[-1] > right[-1]: left, right = right, left
        return merge(left,right[:-1]) + [right[-1]]
    return list(left or right)


def mergesort(seq):
    if len(seq) <= 1:
        return list(seq)
    i = len(seq)//2
    return merge(mergesort(seq[:i]),mergesort(seq[i:]))
```

4. Define a function solve() to solve a classic ancient Chinese puzzle.

We count the number of heads (denoted by numheads) and the number of legs (denoted by numlegs) among the chickens and rabbits in a farm. The solve() function need to return the number of rabbits and the number of chickens. If no solution is found, solve() function returns a string `No solutions!`.

X represents for the number of chickens

Y represents for the number of rabbits

numheads = X+Y

numlegs = 2*X+4*Y

Hint:

Use for loop to iterate all possible solutions.

Suppose you want to return two variables x and y, you can write return x, y

For Example:

Test	Result
print(solve(35,94))	(23,12)
print(solve(30,100))	(10,20)
print(solve(1,3))	'No solutions!'

Answer:

```
def solve(numheads,numlegs):
    ns='No solutions!'
    for i in range(numheads+1):
        j=numheads-i
        if 2*i+4*j==numlegs:
            return i,j
    return ns
```

5. Define a method matches() to determine whether the number of parentheses in the string matches.

For example:

Test	Result
matches(‘(hello(welcome(come to)))’)	True
matches(‘(hello(welcome((come to)))’)	False

Answer:

```
def matches(str):
    left=0
    right=0
    for i in str:
        if i=="(":
            left=left+1
        if i==")":
            right=right+1
    return left==right
```

6. Write a function named f(n) to compute:

$$f(n) = f(n-1) + 100 \; when \; n > 0$$

$$and \; f(0) = 1$$

$$with \; a \; given \; n \; input \; by \; console \; (n > 0)$$

For example:

Test	Result
print(f(0))	1
print(f(1))	101
print(f(4))	401

Answer:

```
def f(n):
    if n==0:
        return 1
    else:
        return f(n-1)+100
```

7. Define a function concat_two_dicts(a,b) that accepts two arguments of type dict such that

- concat_two_dicts(a, b) will return a new dictionary containing all the items in a,
- and the items in b that have different keys than those in a.

For example:

Test	Result
a = {"x": 10, "z": 30} b = {"y": 20, "z": 40} print(concat_two_dicts(a, b)) print(concat_two_dicts(b, a))	{"x": 10, "z": 30, "y": 20} {"x": 10, "z": 40, "y": 20}
a = {"x": 10, "z": 30} b = {"y": 20} print(concat_two_dicts(a, b)) print(concat_two_dicts(b, a))	{"x": 10, "z": 30, "y": 20} {"x": 10, "z": 30, "y": 20}

Answer:

```
def concat_two_dicts(a, b):
  return {**b, **a}
```

8. Define a function checkValue() that accepts a variable number of integer numbers as arguments. checkValue() will count the number of odd number (denoted by x) and the number of even number (denoted by y). Finally, checkValue() will print "There is/are x odd number(s) and y even number(s)".

For example:

Test	Result
checkValue(7)	There is/are 1 odd number(s) and 0 even number(s)
checkValue(7,8,9)	There is/are 2 odd number(s) and 1 even number(s)
checkValue(10,20,30,40)	There is/are 0 odd number(s) and 4 even number(s)

Answer:

```
def checkValue(*args):
    x=0
    y=0
    for i in args:
        if i%2 ==0:
            y+=1
        else:
            x+=1
    print("There is/are {} odd number(s) and {} even number(s)".format(x,y))
```

9. Define a function deduped which

- takes an input list as an argument, and

- returns a copy of the list with all duplicate values removed but orders of remaining items preserved.

For example:

Test	Result
print(deduped([1,2,2,3,4]))	[1,2,3,4]
print(deduped([1,1,2,3,3,4]))	[1,2,3,4]

Answer:

```
def deduped(input_list):
    appeared = set()
    output = []
    for x in input_list:
        if x not in appeared:
            appeared.add(x)
            output.append(x)
        return output
```

10. Define a function shortened which

- takes
  - ➢ an input list input_list as the first argument,
  - ➢ a sequence pos of non-negative indices as the second argument, and
- returns a copy of input_list but with elements indexed by the indices in pos removed.

For example:

Test	Result
print(shortened([12,24,35,70,88,120,155], [0,4,5]))	[24, 35, 70, 155]
print(shortened([12,24,35,70,88,120,155], [1,6]))	[12, 35, 70, 88, 120]

Answer:

```
def shortened(input_list,pos):
    return [input_list[i] for i in range(len(input_list)) if i not in pos]    for x in input_list:
```

-- END OF PAPER --