NATIONAL UNIVERSITY OF SINGAPORE

Department of Computer Science, School of Computing

IT5001 – *Software Development Fundamentals*

Academic Year 2024/2025, Semester 1

Final Assessment

QUESTION BOOKLET

25 November 2024 **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, NUS-approved calculator.

3. Use of any other electronic devices, including smart watches, is **NOT** allowed.

4. This paper comprises **FORTY-SEVEN (47) questions** and **TWENTY-FIVE (25) pages** including this cover page.

5. Do **NOT** write your name on any document you submit.

6. Write your **Student Number (starting with A)** on the 'ANSWER BOOKLET'.

7. Write and shade your **Student Number (starting with A)** on the 'OCR Answer Sheet' using a 2B pencil.

8. For Multiple-Choice Questions (MCQs) Q1 to Q38 (inclusive), write your answers in the empty space beside the question.

9. For the remaining questions, write your answers in the space provided; no extra sheets will be accepted as answers. You may write in blue or black with a pencil or pen.

10. 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.

11. You **cannot** communicate with anyone other than the invigilators throughout the exam.

12. You **must attempt the assessment on your own**. The University takes a zero-tolerance approach towards plagiarism and cheating.

MULTIPLE CHOICE QUESTIONS [50 marks]

*For each question, choose only **ONE (1)** answer.*

1. The following Python expression is entered into a fresh Python shell with no prior import statements. Determine the result from evaluating the following expression. **[1 mark]**

```
11 - 5 ** 2 // 4
```

A. `-4`
B. `4.75`
C. `5`
D. `9`
E. `10`

2. The following Python expression is entered into a fresh Python shell with no prior import statements. Determine the result from evaluating the following expression. **[1 mark]**

```
6 - - 8 < 0 or 5 % 3
```

A. `True`
B. `False`
C. `1`
D. `2`
E. Evaluating this expression yields an error

3. The following Python expression is entered into a fresh Python shell with no prior import statements. Determine the result from evaluating the following expression. **[1 mark]**

```
0 or not 1 / (float(4//3)-4/3) and 1 / (int(4/3)-4//3)
```

A. `True`
B. `False`
C. `1`
D. `0`
E. Evaluating this expression yields an error

4. The following Python expression is entered into a fresh Python shell with no prior import statements. Determine the result from evaluating the following expression. **[1 mark]**

```
'lion'[1] + 'witch'[2:3] + 'wardrobe'[4:]
```

A. `'itr'`
B. `'itcr'`
C. `'itrobe'`
D. `'itcrobe'`
E. `'iontrobe'`

5. The following Python expression is entered into a fresh Python shell with no prior import statements. Determine the result from evaluating the following expression. **[1 mark]**

```
'Secret Gardens'[9::-1]
```

   A. `'raG terceS'`
   B. `'rdens'`
   C. `'snedr'`
   D. `'' # Empty string`
   E. Evaluating this expression yields an error

6. The following Python expression is entered into a fresh Python shell with no prior import statements. Determine the result from evaluating the following expression. **[1 mark]**

```
('xmas' * 3)[-8:-2][-1]
```

   A. `'x'`
   B. `'m'`
   C. `'a'`
   D. `'s'`
   E. `'' # Empty string`

7. The following Python expression is entered into a fresh Python shell with no prior import statements. Determine the result from evaluating the following expression. **[1 mark]**

```
[[7, 11], 42][0] + [2, 4]
```

   A. `[2, 4]`
   B. `[7, 2, 4]`
   C. `[7, 11, 2, 4]`
   D. `[[7, 11], 2, 4]`
   E. Evaluating this expression yields an error

8. What is the output of the following code? **[1 mark]**

```
ls1 = [['p', 'q'], 'x', ['p', 'q']]
ls2 = ls1[1:]
ls2[1][1] = 's'
print(ls1)
```

   A. `[['p', 'q'], 'x', ['p', 'q']]`
   B. `[['p', 'q'], 'x', ['p', 's']]`
   C. `[['p', 's'], 'x', ['p', 's']]`
   D. `['x', ['p', 's']]`
   E. Executing this code yields an error

9. The following Python expression is entered into a fresh Python shell with no prior import statements. Determine the result from evaluating the following expression. **[1 mark]**

```
(('R', 255), ('G', 0), ('B', 255))[1]
```

A. `'G'`
B. `255`
C. `('G', 0)`
D. `(255,)`
E. `('R', 255)`

10. What is the output of the following code? **[1 mark]**

```
tup1 = (1, 2, (3,))
print(tup1.append(9))
```

A. `(1, 2, 3, 9)`
B. `(1, 2, (3, 9))`
C. `(1, 2, (3,), 9)`
D. `None`
E. Executing this code yields an error

11. The following Python expression is entered into a fresh Python shell with no prior import statements. Determine the result from evaluating the following expression. **[1 mark]**

```
set('hello world') - set('word')
```

A. `{'h', 'e', 'l', ' '}`
B. `{'h', 'e', 'l', 'l', 'o', ' ', 'l'}`
C. `{'h', 'e', 'l'}`
D. `'hello l'`
E. `{'hello l'}`

12. The following Python expression is entered into a fresh Python shell with no prior import statements. Determine the result from evaluating the following expression. **[1 mark]**

```
{(0,0), (1,1)} | {0,1} & {(0,0), 1}
```

A. `{0, (1, 1)}`
B. `{(0, 0), (1, 1)}`
C. `{1, (0, 0)}`
D. `{1, (1, 1), (0, 0)}`
E. Evaluating this expression yields an error

13. The following Python expression is entered into a fresh Python shell with no prior import statements. Determine the result from evaluating the following expression. **[1 mark]**

```
{2: 4, 3: 9, 5: 25}.get(4, 0)
```

A. 0
B. 2
C. 5
D. None
E. Evaluating this expression yields an error

14. What is the output of the following code? **[1 mark]**

```
d = {1: 'en', 2: 'zh', 3: 'ms', 4: 'ta'}
d.pop(1)
print(d[1])
```

A. en
B. zh
C. False
D. None
E. Executing this code yields an error

15. What is the output of the following code? **[1 mark]**

```
me = {'IT5001': {'marks': {'pe': 65, 'midterm': 72}, 'final': True}}
hack = me['IT5001']['marks'].copy()
hack['pe'] = 99
hack['final'] = None
me['IT5001']['marks'] = hack
print(me.get('IT5001'))
```

A. {'marks': {'pe': 99, 'midterm': 72, 'final': None}, 'final': True}
B. {'marks': {'pe': 99, 'final': None}, 'final': True}
C. {'marks': {'pe': 99, 'midterm': 72}, 'final': None}
D. {'marks': {'pe': 99, 'midterm': 72, 'final': None}}
E. {'marks': {'pe': 65, 'midterm': 72}, 'final': True}

16. What is the output of the following code? **[1 mark]**

```
sun, wind, rain = 0, 1, 1
if rain:
    if wind:
        print('Cold')
    print(rain)
elif sun or wind:
    print('Sweat')
```

A.
```
Cold
1
Sweat
```

B.
```
Cold
Sweat
```

C.
```
Cold
1
```

D.
```
Cold
True
```

E. No output is produced

17. Given the following function:

```
def foo(x):
    if x > 100:
        print('Too much!')
        return
    if x % 10:
        print('OK')
    elif x % 3:
        print('KO')
```

Which value of `x` will cause `foo(x)` to print nothing (i.e., empty output)? **[1 mark]**

A. 20
B. 30
C. 33
D. 100
E. Any value of `x` will cause `foo(x)` to print something

18. Given the following function that takes in two lists:

```
def bar(la, lb):
    if not la and not lb:
        return []
    out = []
    if len(la) > len(lb):
        out.append(la[0])
    elif lb:
        out.append(lb[0])
    else:
        print('Pass')
    return out
```

Which of the following statements about `bar(la, lb)` is false? **[1 mark]**

A. The function is equivalent to `(lb+la)[0]` whenever `lb` has more elements than `la`
B. The function will never print `'Pass'`
C. The list `out` is guaranteed to be non-empty when it is returned at the end of the function
D. The returned list has at most one element
E. None of the above statements are false

19. What is the output of the following code? **[1 mark]**

```
i, x = 0, 20
count = 0
while i < x:
    i += 2
    count += 1
print(count)
```

A. `0`
B. `9`
C. `10`
D. `19`
E. `20`

20. What does the following function return? **[1 mark]**

```
def foo(tup, val):
    count = 0
    for e in tup:
        if e >= val:
            continue
        count += 1
    return count
```

A. The index of `val` in `tup`
B. `1 +` the index of `val` in `tup`
C. The number of elements in `tup`
D. The number of elements in `tup` that are smaller than `val`
E. The number of elements in `tup` that are larger than `val`

21. How many lines are printed by the following code? **[1 mark]**

```
for p in range(5):
    for q in range(p):
        print(p+q)
```

A. 9
B. 10
C. 11
D. 15
E. 21

22. Given that `n` is a positive integer, what does the following function return? **[1 mark]**

```
def rec(n):
    if n == 1:
        return n
    return n + rec(n-1)
```

A. The value of `n * n`
B. The value of `n * (n-1)`
C. The value of `n + n-1`
D. The sum of integers from `1` to `n-1` (inclusive)
E. The sum of integers from `1` to `n` (inclusive)

23. Consider the following code that calls the modified magic function from tutorial (qSort):

```
def magic(lst):
    if not lst:
        return lst
    print('Abracadabra!')
    pivot = lst[0]
    lsta = magic([i for i in lst if i < pivot])
    lstb = magic([i for i in lst if i > pivot])
    return lsta + [pivot] + lstb

magic([22, 6, 18, 39, 1])
```

Which of the following statements about `bar(la, lb)` is <u>false</u>? **[1 mark]**

A. 1
B. 3
C. 4
D. 5
E. 6

24. Given the following function definition:

```
def foo(a):
    return a + 2
```

Which lambda expression is equivalent to calling `foo(9)`? **[1 mark]**

A. `(lambda a: a + 2)(9)`
B. `lambda a: a + 2`
C. `lambda a: 9 + 2`
D. `lambda(a = 9): a + 2`
E. `lambda(9): 9 + 2`

25. What is the output of the following code? **[1 mark]**

```
def bar(f, a, b):
    return f(a, b)

a, b = 2, 3
x = bar(lambda a,b:a*b, bar(lambda a,b:a+b, a, b), bar(lambda a,b:a-b, a, b))
print(x)
```

A. `5`
B. `-5`
C. `6`
D. `-6`
E. `-1`

26. What is the output of the following code? **[1 mark]**

```
class Quote():
    def __init__(self, name, quote):
        self.name = name
        self.say = name + ': ' + quote

teacher = Quote('Alan', 'Code, pray, love')
print(teacher.say)
```

A. `Alan`
B. `Code, pray, love`
C. `Alan: Code, pray, love`
D. `Code, pray, love: Alan`
E. Executing this code yields an error

27. What is the output of the following code? **[1 mark]**

```
class Item():
    def __init__(self, name, price, qty):
        self.name = name
        self.price = price
        self.qty = qty
        self.revenue = 0

    def discount(self, discount):
        if discount < self.price:
            self.price -= discount

    def buy(self):
        if self.qty > 0:
            self.qty -= 1
            self.revenue += self.price

labubu = Item('labubu', 300, 3)
labubu.discount(100)
labubu.buy()
labubu.discount(250)
labubu.buy()
labubu.discount(50)
labubu.buy()
labubu.buy()
print(labubu.revenue)
```

A. 450
B. 500
C. 550
D. 700
E. 800

28. What is the output of the following code? **[1 mark]**

```
class Person():
    def __init__(self, name):
        self.name = name
    def intro(self):
        print('Hi, this is ' + self.name)

class Adult(Person):
    def __init__(self, name):
        super().__init__(name)
        self.spouse = None
    def marry(self, spouse):
        self.spouse = spouse

rachel = Adult('Rachel')
rachel.marry(Adult('Ross'))
rachel.spouse.intro()
```

A. `Hi, this is Ross`
B. `Hi, this is Rachel`
C. `None`
D. Executing this code yields an error
E. No output is produced

29. What is the output of the following code? **[1 mark]**

```
class A():
    def __init__(self, name):
        self.name = name
    def whoami(self):
        print(self.name)

class B(A):
    def whoami(self):
        print('B')
        super().whoami()

class C(B):
    def whoami(self):
        super().whoami()
        print('C')

c = C('Peter')
c.whoami()
```

A.
```
B
Peter
C
```

B.
```
Peter
C
```

C.
```
Peter
B
C
```

D.
```
C
B
Peter
```

E.
```
C
```

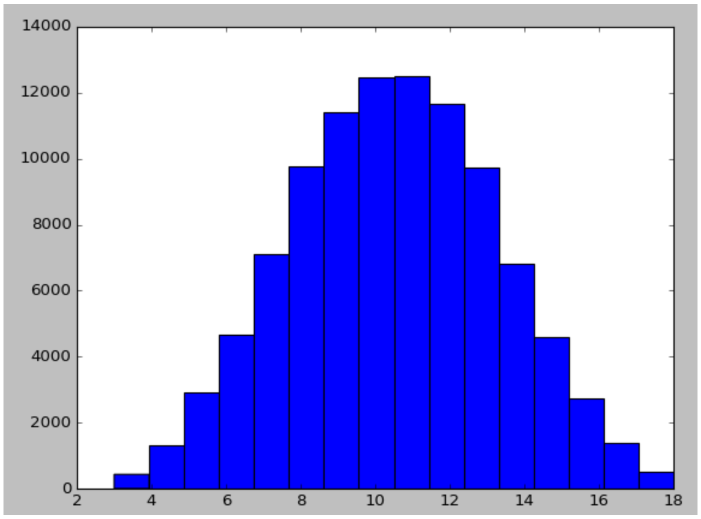
30. What package mentioned in class is for image processing? **[1 mark]**

A. IMAGEPL
B. IMAGEIOLIB
C. PSLIB
D. PIL
E. PILLOWS

31. What is the meaning of "EXIF" as mentioned in class? **[1 mark]**

    A. Assertion
    B. A branch in the `try` block
    C. A type of exception
    D. Extra conditional statements
    E. Information about an image

32. In the package `matplotlib.pyplot`, which of the following functions can produce a graph like as shown? **[1 mark]**



    A. `bar()`
    B. `barh()`
    C. `hist()`
    D. `pie()`
    E. `plot()`

33. What is the output of the following code? **[3 marks]**

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

def g(n):
    if n == 0:
        return 1
    else:
        return 1+f(n-2)

print(f(10))
```

A. `11`
B. `10`
C. `9`
D. `8`
E. `6`

34. What is the output of the following code? **[3 marks]**

```
data = {'a': 10, 'b': 12, 'c': 30}
for key, value in list(data.items()):
    if value % 10 == 0:
        data[key] = value // 2
    else:
        data[key + '1'] = value * 2
        del data[key]

total = sum(data.values())
print(total)
```

A. `59`
B. `52`
C. `44`
D. `26`
E. None of the above

35. What is the output of the following code? **[3 marks]**

```
lst = [9,12,3,7,11,2,4,8,5]
left = 1
right = len(lst)-1
while left < right:
    while lst[left] < lst[0]:
        left += 1
    while lst[right] > lst[0]:
        right -= 1
    if left < right:
        lst[left],lst[right] = lst[right],lst[left]

print(lst)
```

A. `[2, 3, 4, 5, 7, 8, 9, 11, 12]`
B. `[5, 3, 7, 8, 2, 4, 9, 11, 12]`
C. `[9, 5, 3, 7, 8, 2, 4, 11, 12]`
D. `[12, 11, 9, 8, 7, 5, 4, 3, 2]`
E. None of the above

36. What is the output of the following code? **[3 marks]**

```
arr = [5, 2, 9, 1, 8]
n = len(arr)
for i in range(n):
    flag = False
    for j in range(0, n-i-1):
        if arr[j] > arr[j+1]:
            flag = True
            arr[j], arr[j+1] = arr[j+1], arr[j]
    if not flag:
        break

print(arr)
```

A. `[1, 2, 5, 8, 9]`
B. `[2, 1, 5, 8, 9]`
C. `[2, 5, 1, 8, 9]`
D. `[5, 2, 9, 1, 8]`
E. `[9, 8, 5, 2, 1]`

37. What is the output of the following code? **[3 marks]**

```
def f(seq):
    if seq == []:
        return seq
    elif type(seq) != list:
        return [seq]
    else:
        return f(seq[1:]) + f(seq[0])

lst = [1,2,[3,4],[5,[6,7]]]
print(f(lst))
```

A. `[1, 2, 3, 4, 5, 6, 7]`
B. `[1, 2, 3, 4, 5, [6, 7]]`
C. `[1, 2, [3, 4], [5, [6, 7]]]`
D. `[7, 6, 5, 4, 3, 2, 1]`
E. `[[[7, 6], 5], [4, 3], 2, 1]`

38. What is the output of the following code? **[3 marks]**

```
def f(seq):
    if not seq:
        return seq
    if type(seq) != list:
        return seq
    return [f(seq[0])]*2 + f(seq[1:])

lst = [1,[2]]
print(f(lst))
```

A. `[1, 2, 2]`
B. `[1, 1, 2, 2]`
C. `[1, 1, [2, 2], [2, 2]]`
D. `[2, [4, 4]]`
E. `[2, [4]]`

FILL-IN-THE-BLANKS QUESTIONS [48 marks]

For each question, replace each blank with a valid Python expression/statement. Note that your code needs to be syntactically correct to gain marks. You cannot use any semicolon (;) and we will deduct marks if your answer is too long.

39. Given below is an incomplete implementation of a function that prints out 10 multiples of the input `n`. Replace each blank with a valid Python expression/statement. **[6 marks]**

```
def print_10_multiples(n):
    for i in range(___BLANK_1___):
        print(___BLANK_2___)
```

Here is an example:

```
>>> print_10_multiples(3)
3
6
9
12
15
18
21
24
27
30
```

BLANK	ANSWER
1	
2	

40. Given a list `L` of integers, function `sum_of_squares()` returns the sum of the squares of each integer. e.g., if `L = [1,2,10]`, the sum of squares will be `1*1 + 2*2 + 10*10 = 105`.

Complete the function by replacing each blank with a valid Python expression/statement. **[6 marks]**

```
def sum_of_squares(L):
    total = 0
    for i in ___BLANK_1___:
        total = ___BLANK_2___
    return total
```

Here is a sample run:

```
>>> print(sum_of_squares([2,3,10]))
113
```

BLANK	ANSWER
1	
2	

41. Given a list `L` of integers, complete function `mul_list()` with valid Python expressions/statements such that it returns a list that contains all the integers in `L` multiplied by `n`. **[6 marks]**
e.g., if `L = [1,3,9]` and `n = 4`, the function should output `[4,12,36]`.

```
def mul_list(L,n):
    return ___BLANK_1___
```

Here is a sample run:

```
>>> print(mul_list([1,3,9,2,4,7],2))
[2, 6, 18, 4, 8, 14]
```

BLANK	ANSWER
1	

42. Given a list `L` of integers, complete function `abs_diff()` with valid Python expressions/statements such that it returns a list of the absolute difference between every two consecutive integers in `L`. **[6 marks]**

```
def abs_diff(L):
    output = []
    for i in ___BLANK_1___:
        output.append(___BLANK_2___)
    return output
```

Here is a sample run:

```
>>> print(abs_diff([1,2,4,9,3,1]))
[1, 2, 5, 6, 2]
```

BLANK	ANSWER
1	
2	

43. Given an input string s and a dictionary d such that all the keys and values are single alphabet characters in uppercase, complete function encrypt() using valid Python expressions/statements such that it changes all the characters in s from the keys to the values in d. If a character c is not a key of d, that character remains unchanged. **[6 marks]**

```
def encrypt(d,s):
    output = ''
    for c in ___BLANK_1___:
        if ___BLANK_2___:
            output = ___BLANK_3___
        else:
            output = ___BLANK_4___
    return output
```

Here is a sample run:

```
>>> print(encrypt({'A':'C','Q':'E','N':'M'},"BANANA"))
BCMCMC
>>> print(encrypt({'P':'C','T':'E','N':'M'},"PYTHONIDLE"))
CYEHOMIDLE
```

BLANK	**ANSWER**
1	
2	
3	
4	

44. Let input s be a string with uppercase alphabet characters. Complete function count_alpha() using valid Python expressions/statements so that it returns a dictionary whose keys are all the characters in s, and the corresponding values are the number of times the characters appear in s. **[6 marks]**

```
def count_alpha(s):
    output = dict()
    for c in ___BLANK_1___:
        if ___BLANK_2___:
            output[c] = ___BLANK_3___
        output[c] = ___BLANK_4___
    return output
```

For example,

```
>>> print(count_alpha("QUEUE"))
{'Q': 1, 'U': 2, 'E': 2}
>>> print(count_alpha("BANANAS"))
{'B': 1, 'A': 3, 'N': 2, 'S': 1}
```

BLANK	ANSWER
1	
2	
3	
4	

45. Given a list `L` of strings, complete function `double_reverse()` with valid Python expressions/statements such that it returns a list containing the reverse of all strings in `L`, where they are also in reverse order from `L`. **[6 marks]**

```
def double_reverse(L):
    return ___BLANK_1___
```

For example,

```
>>> print(double_reverse(['ti', 'od', 'uoy', 'nac']))
['can', 'you', 'do', 'it']
```

BLANK	ANSWER
1	

46. Given a 2D "image" m, complete function double_image() with valid Python expressions/statements such that it scales the image by a factor of 2, so that every "pixel" in m becomes a 2x2 "larger pixel" in the output. **[6 marks]**

You are given a function czm(r,c) which is the same as create_zero_matrix(r,c) from lecture.

```
def double_image(m):
    r = len(m)
    c = len(m[0])
    output = ___BLANK_1___
    for i in range(___BLANK_2___):
        for j in range(___BLANK_3___):
            output[i][j] = ___BLANK_4___
    return output
```

Here is a sample run:

```
>>> m2 = [['.','#','.'],['#','#','#'],['.','#','.']]
>>> pprint(double_image(m2))
[['.', '.', '#', '#', '.', '.'],
 ['.', '.', '#', '#', '.', '.'],
 ['#', '#', '#', '#', '#', '#'],
 ['#', '#', '#', '#', '#', '#'],
 ['.', '.', '#', '#', '.', '.'],
 ['.', '.', '#', '#', '.', '.']]
```

BLANK	ANSWER
1	
2	
3	
4	

<u>FREE OPINION QUESTION [2 marks]</u>

47. What is the most interesting thing you learnt in this course? Or, if you can rewind time and go back to the beginning of the semester, what will you do differently in this course? **[2 marks]**

-- End of Assessment --

NATIONAL UNIVERSITY OF SINGAPORE

Department of Computer Science, School of Computing

IT5001 – *Software Development Fundamentals*

Academic Year 2024/2025, Semester 1

Final Assessment

SOLUTIONS MANUAL

25 November 2024 **Time allowed:** 2 hours

MULTIPLE CHOICE QUESTIONS [50 marks]

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

FILL-IN-THE-BLANKS QUESTIONS [48 marks]

NOTE: *The following are only ideas for solving these questions; better solutions exist, and if your code follows a different approach but still returns the correct output (and within 50 characters or less per blank), you may consider your solution correct.*

39.

```
def print_10_multiples(n):
    for i in range(1, 11)      # BLANK 1
        print(n*i)             # BLANK 2
```

40.

```
def sum_of_squares(L):
    total = 0
    for i in L:                # BLANK 1
        total = total + i*i    # BLANK 2
    return total
```

41.

```
def mul_list(L,n):
    return [i*n for i in L]    # BLANK 1
```

42.

```
def abs_diff(L):
    output = []
    for i in range(len(L)-1):               # BLANK 1
        output.append(abs(L[i]-L[i+1]))     # BLANK 2
    return output
```

43.

```
def encrypt(d,s):
    output = ''
    for c in s:                         # BLANK 1
        if c in d.keys():               # BLANK 2
            output = output + d[c]      # BLANK 3
        else:
            output = output + c         # BLANK 4
    return output
```

44.

```
def count_alpha(s):
    output = dict()
    for c in s:                         # BLANK 1
        if not c in output:             # BLANK 2
            output[c] = 0               # BLANK 3
        output[c] = output[c] + l       # BLANK 4
    return output
```

45.

```
def double_reverse(L):
    return [c[::-1] for c in L][::-1] # BLANK 1
```

46.

```
def double_image(m):
    r = len(m)
    c = len(m[0])
    output = czm(r*2,c*2)                   # BLANK 1
    for i in range(r*2):                    # BLANK 2
        for j in range(c*2):                # BLANK 3
            output[i][j] = m[i//2][j//2]    # BLANK 4
    return output
```

-- End of Solutions Manual --