Massachussets Institute of Technology

Neuroscience of programming – Python fluency test

Contact: Anna Ivanova (annaiv@mit.edu), Shash Srikant (shash@mit.edu)

Name: ______________________________

E-mail ID: ______________________________
*If you have an `@mit.edu` email ID, please fill in that.

- This exam contains 14 pages (including this cover page) and 23 questions. Unless specified, assume Python version > 3.6.0. We do not however assess knowledge of specific libraries.
- Please answer within the box/space provided. Only those will be considered your final answers.
- There are three types of questions on this test. One, which asks you to write out the expected output of a program. Two, which asks you to fix an error in a program. Three, which asks you to design an algorithm and implement it in Python.
- For questions requiring you to fix the error,
  - you do not have to re-write the entire fixed program. Only rewrite lines containing your fixes.
  - there could be multiple ways to answer these questions. Answering any one is fine.
  - do not replace the entire code snippet with an alternate approach. Try fixing the provided lines as much as possible.
- All questions are weighted equally.
- If you're unable to answer any question but have an idea about it, write it down in the box provided. Likewise, if you think any question is ambiguous, write down your concern.

Good luck and happy Pythoning!

1. What does the following code output?

```
min1, max1 = 0, 100
min2, max2 = 200, 300
num1 = min1*max2
num2 = min2 + max1 + num1
print(num2)
```

2. What does the following code output?

```
numbers = [10, 20, 30]
position = 2
for index in range(len(numbers)):
    position = position + index

print(position)
```

3. What does the following code output?

```
line = "abc"
letters = "p"

for char in line:
    letters = letters[-1] + char

print(letters)
```

4. What does the following code output?

```
h, m = 100, 200
h_deg, m_deg = h//2, m//3
# In Python3, // rounds down to the nearest whole number
angle = abs(h_deg - m_deg)

if angle > 180:
    angle = 360 - angle

print(int(angle))
```

5. What does the following code output?

```
movie, n = "Shrek", 2
first_part = movie[:n]
last_part = movie[(n+1):]
print(first_part + last_part)
```

6. What does the following code output?

```
target = ['s', 'i', 'r', 'e', 'n']
template = ['c', 'a', 't']

for index in range(len(template)):
    template[index] = target[index]

print(template)
```

7. What does the following code output?

```
s =[1,2,3]
print(s[1:5])
```

8. What does the following code output?

```
n = 4
a = [[2] * i + [1] + [0] * (n - i - 1) for i in range(n)]

for row in a:
    print(' '.join([str(elem) for elem in row]))
```

9. What does the following code output?

```
def extendList(val, lst=[]):
    lst.append(val)
    return lst

list1 = extendList(10)
list2 = extendList(123,[])
list3 = extendList('a')

print("list1 = %s" % list1)
print("list2 = %s" % list2)
print("list3 = %s" % list3)
```

10. What does the following code output?

```
list1 = ['1','2','3']
list3 = list1
print(list1 is list3)
print(list1 == list3)
print(id(list1) == id(list3))
```

11. What does the following code output?

```
def foo(param1, *param2):
    print(param1)
    print(param2)
def bar(param1, **param2):
    print(param1)
    print(param2)
foo(1,2,3,4,5)
bar(6,a=7,b=8)
```

12. What does the following code output?

```
matrix=[(1,2,3),(4,5,6),(7,8,9),(10,11,12)]
result = zip(*matrix)
for row in result:
    print(row)
```

13. You are given two files - `one.py` and `two.py`. `two.py` is in the same directory as `one.py` and imports it. You run `two.py`. What output do you see?

```python
## file one.py
def func():
  print("1") # func() in one.py

print("2") # top-level in one.py

if __name__ == "__main__":
  print("3") # one.py is being run directly
else:
  print("4") # one.py imported into another module


## file two.py
import one
print("5") # top-level in two.py
one.func()

if __name__ == "__main__":
  print("6") # two.py is being run directly
else:
  print("7") # two.py imported into another module
```

14. What does the following code output?

```python
import copy
a = [1,2,3,4]
b = copy.copy(a) # Shallow copy

print(a)
print(b)

b.append(1)

print(a)
print(b)
```

15. While implementing the following script, you see the error mentioned below. Fix the error. Rewrite code in the empty space on the right.

```
Traceback (most recent call last):
  File "test.py", line 13, in <module>
    me.speak()
TypeError: speak() takes 0 positional arguments but 1 was given
```

```python
1  class Person:
2  
3    def __init__(self, first_name, last_name):
4      self.first = first_name
5      self.last = last_name
6  
7    def speak():
8      print("My name is  "+ first + " " + last)
9  
10 me = Person("Jane", "Goodall")
11 you = Person("Bob", "Sapolsky")
12 
13 me.speak()
```

16. While implementing the following script, you see the error mentioned below. Fix the error. Rewrite code in the empty space on the right.

```
Traceback (most recent call last):
  File "test.py", line 7, in <module>
    word1[i] = space;
TypeError: 'str' object does not support item assignment
```

```
word = input("Enter two words (
   ↪ space separated): ")
word1 = word
number = len(word)
space = '/'
for i in range(number):
    if(word1[i] == " "):
        word1[i] = space
print(word1)
```

17. Your friend has written a simple game which accepts integers between 1 and 10. Running her script generates the following error. Fix the error. Rewrite code in the empty space on the right.

```
Traceback (most recent call last):
File "test.py", line 3, in <module>
  if (choice + random.randint(1,10))%10 == random.randint(1,10):
TypeError: Cannot convert 'int' object to str implicitly
```

```
import random
choice = input("Enter your guess
   ↪ between 1 and 10: ")
if (choice + random.randint(1, 10))%10
   ↪  == random.randint(1, 10):
    print("You guessed right!")
else:
    print("Tough luck.")
```

18. While implementing the following script, you see the error mentioned below. Fix the error. Rewrite code in the empty space on the right.

```
Traceback (most recent call last):
  File "test.py", line 14, in <module>
    uq_pairs = set(lst_pairs)
TypeError: unhashable type: 'list'
```

```
lst_pairs = []

with open('file.txt','r') as fp:
    lines = fp.readlines()
    fp.close()

for i in lines:
# each line has two strings,
# separated by a colon (:)
# e.g. check:true
    lst_pairs.append(i.split(":"))

# Find unique set of entries
uq_pairs = set(lst_pairs)
```

19. You have two files - `base.py` and `inherit.py`. Running `inherit.py` results in the following error. Fix the error. Rewrite code in the empty space on the right.

```
Traceback (most recent call last):
  File "inherit.py", line 3, in <module>
    class Visitor(base):
TypeError: module.__init__() takes at most 2 arguments (3 given)
```

```
## base.py
class Parent():
    def __init__(self):
        print('in init')


## inherit.py
import base
class Visitor(base):
    pass
instance = Visitor()
```

20. Alice writes a method to calculate the lowest positive number in a list of integers. On running it on a simple test case like `[16, -6, 1, 6, 6]`, she sees the following error. Fix the error. Rewrite code in the empty space on the right.

```
File "test.py", line 15, in <module>
    print(find_lowest(a))
File "test.py", line 12, in find_lowest
  return lowest(lst[0], lst[1:])
File "test.py", line 9, in lowest
  return lowest(rest[0], rest[1:])
File "test.py", line 9, in lowest
  return lowest(rest[0], rest[1:])
File "test.py", line 13, in lowest
  return lowest(first, rest)
File "test.py", line 13, in lowest
  return lowest(first, rest)
[Previous line repeated 993 more times]
File "test.py", line 6, in lowest
  if len(rest) == 0:
RecursionError: maximum recursion depth exceeded in comparison
```

```
1  def find_lowest(lst):
2   # Return the lowest positive
3   # number in a list.
4   def lowest(first, rest):
5      # Base case
6      if len(rest) == 0:
7        return first
8
9      if first > rest[0] or first < 0:
10       return lowest(rest[0], rest[1:])
11
12     else:
13       return lowest(first, rest)
14
15  return lowest(lst[0], lst[1:])
16
17 a = [16, -6, 1, 6, 6]
18 print(find_lowest(a))
```

21. You want to design a class in a way that it can dynamically call one of `function1` or `function2`. You have an initial attempt at making it work, but it compiles to the error mentioned below. Fix method `get` to achieve the desired functionality.

```
Traceback (most recent call last):
  File "test.py", line 18, in <module>
    obj.get()
  File "test.py", line 7, in get
    self.func_name()
AttributeError: 'MyClass' object has no attribute 'func_name'
```

```python
class MyClass:
  def __init__(self, i):
    self.i = i

  def get(self):
    func_name = 'function'+str(self.i)
    self.func_name()

  def function1(self):
    print("Hello")

  def function2(self):
    print("There")

obj = MyClass(1)
obj.get()
```

22. The following script invokes the method `ui()`. A bug in it produces this error when provided a certain sequence of inputs. Assume methods `readcache` and `writecache` are appropriately defined. Fix the error. Rewrite code in the empty space on the right.

```
Traceback (most recent call last):
  File "test.py", line 30, in <module>
    zipname = x[0]
TypeError: 'NoneType' object is not subscriptable
```

```
 1 def read_cache():
 2  # Perform some read operations
 3  # Return lst = [zipname, direc]
 4  return lst
 5
 6 def write_cache(a, b):
 7  # Perform some write operations
 8  # Nothing to return
 9
10 def ui():
11  g = input("Use cache? (y/n):")
12
13  if g == "y":
14    i = read_cache()
15    return i
16
17  elif g == "n":
18    zipname = input("Zip fname:")
19    direc = input("Zip directory:")
20    write_cache(zipname, direc)
21    i = [zipname, direc]
22    return i
23
24  else:
25    print("Respond with y/n \n")
26    ui()
27 # End of ui()
28
29 x = ui()
30 zipname = x[0]
31 direc = x[1]
```

23. You are given a grid of `1`s and `0`s. Your job is to find out the number of *clusters* of `1`s in it. A `1` belongs to a *cluster* if it has a `1` in either of the four positions - above, below, to its right, or to its left. A `1` surrounded by 0s in the four positions represents a *cluster* by itself. For example,

```
0 0 0 1
1 0 0 0
0 1 0 1
```

has 4 *clusters*, with each isolated `1` forming one cluster, while

```
1 1 1 1
1 0 0 0
0 1 1 1
```

has 2 *clusters*.

You will be given a two-dimensional matrix containing `1`s and `0`s. You are required to return an integer denoting the number of *clusters* in the matrix.

```
def find_number_of_clusters(arr):
```