This quiz has been regraded; your score was affected.

Midterm 1-Sp25

- Due Feb 24 at 8pm
- Points 100
- Questions 19
- Available Feb 24 at 8am - Feb 24 at 8pm 12 hours
- Time Limit 50 Minutes

Instructions

Instructions:

- You have **50** minutes to complete the exam.
- The number of points associated with each question is indicated.
- You may **ONLY** use class notes and slides. Use of external notebooks and similar resources is **not allowed.**

Good Luck!

This quiz was locked Feb 24 at 8pm.

Attempt History

	Attempt	Time	Score	Regraded
LATEST	Attempt 1	43 minutes	92 out of 100	100 out of 100

Score for this quiz: 100 out of 100
Submitted Feb 24 at 3:51pm
This attempt took 43 minutes.

Question 1
0 / 0 pts

Academic Integrity

Code of Student Conduct

“The Chancellor may impose discipline for the commission or attempted commission (including aiding or abetting in the commission or attempted commission) of the following types of violations by students, as well as such other violations as may be specified in campus regulations:

102.01 Academic Misconduct: All forms of academic misconduct including but not limited to cheating, fabrication, plagiarism, or facilitating academic dishonesty.”
Collaboration is important and a powerful way to learn. I encourage you to study together and to work

lab assignments together. No matter your academic background, you will learn more if you work alongside others. Ask questions, answer questions, and share ideas.

However, it is important to keep in mind the limits to collaboration. Any work you turn in must be 100% your own. You must not email, post, or share your work with others, and you may not copy the work of another student or from any other source. More generally, you may not submit work that is not your own, or claim jointly authored or copied work to be your own. We use special software to check all submitted work. Copying answers from any source is unacceptable and will be penalized. In addition, posting course content on any third party websites is considered academic misconduct. Any misconduct incident could be reported to the Center for Student Conduct.

Be honest! Cheating is disrespectful, destroys trust, and hurts you both by learning less and via UC penalties. It also takes time from us that we'd rather spend mentoring. What to expect if cheating:

- For 1st incident: Minus 15% of E7 points, write a reflection and reported to campus (risks suspension, expulsion, etc.)

- For 2nd incident: F in E7, and again reported to campus (risks suspension, expulsion, etc.)

Deadlines can be stressful, and we know that under extreme pressure, it becomes tempting to start rationalizing actions that you would otherwise consider inappropriate. Please don't do this. We care that you learn the material, and that your grade reflects your learning. If you feel that you need to engage in academic misconduct to meet a deadline, please reach out to course staff.

Exams

Collaboration is not allowed on any of the exams. Anyone involved in misconduct on any exam will receive an F in the course.

Correct!

- (●) I, hereby, acknowledge that I am not engaging in any academic misconduct.

Question 2

3 / 3 pts

What is the correct way to create a variable named "x" in Python?

- ( ) x == 10
- ( ) x : 10

Correct!

- (●) x = 10
- ( ) int x = 10
- ( ) var x = 10
- ( ) def x = 10

Question 3

3 / 3 pts

What is the output of the following code?

```
z="Python 3.7"
print(z[1])
```

- Error
- P
- 3
- n

Correct!

- y
- Python 3.7
- t
- o

Question 4

3 / 3 pts

What is the output of the following code?

```
print(3 + 3 ** 2)
```

- '3 + 3 ** 2'
- 36
- Error
- 9
- 6

Correct!

- 12

Question 5

3 / 3 pts

What is the output of the following code?

```
z="Python 3.7"
print(z[-3])
```

- h
- Python 3.7
- o
- .
- t
- 7

Correct!

- 3

Question 6

3 / 3 pts

Which of the following is best suited to iterate over a data structure in Python?

- repeat-until loop

Correct!

- for loop
- while loop
- recursion loop
- do-while loop
- if loop

Question 7

3 / 3 pts

What is the output of the following code?

```
e = '5'
f = 5
print (e + f)
```

- 55

Correct!

- Error
- 10
- '5' + '5'
- '5 + 5'

Question 8

3 / 3 pts

What is the output of the following code?

```
print(2 + 3 ** 3)
```

- Nothing

Correct!

- 29
- 75
- 11
- Error
- '2 + 3**3'
- 15

Question 9

3 / 3 pts

What is the output of the following code?

```
b = 3
c = b
b = 1
b - c
print(b)
```

- 3
- True
- Error

Correct!

- 1
- 2
- False

Question 10

3 / 3 pts

What is the output of the following code?

```
test=[1,0,1]
test= test * 2
print(test)
```

- [2,0,2]
- [0,1,1,1,0,1]
- [2,0,2,2,0,2]
- [1,0,1]

Correct!

- [1,0,1,1,0,1]
- [1,0,1,0,1,1]

Question 11
3 / 3 pts

What is the output of the following code?

```
print(4 + 5 ** 2)
```

Correct!

- 29
- 14
- '4 + 5 ** 2'
- 11
- Error
- 81

Question 12
15 / 15 pts

The function below computes the net force acting on a box that is being pulled horizontally, where **F** is the pulling force, **m** is the mass of the box, **g** is the gravitational acceleration and **f** is the coefficient of friction.

Hint: Since g is defined as a float, the output of the function will always be a float unless an external operation is applied.

```
def netForce(F, m, g=9.81, f=0):
    net = F - m * g * f
    return net
```

What is the output of the following expression? If an error occurs, write "Error". If nothing is displayed, write "Nothing". If the evaluation would run forever, write "Forever".

a)

```
print(int(netForce(10,1,f=1)))
```

0

b)

```
print(round(netForce(10,1,f=1),2))
```

0.19

c)

```
m = 10
P = 1
print(netForce(m, P))
```

10.0

d)

```
netForce(10,1,1,1)
print(net)
```

Error

e)

```
m = 1
g = 10
f = 0.5
f = netForce(f, m, g)
print(f)
```

0.5

Answer 1:

Correct! 0

Correct!

0

Answer 2:

Correct! 0.19

Correct!

0.19

Answer 3:

Correct! 10.0

Correct!

10.0

Answer 4:

Correct! Error

Correct Answer

"Error"

Correct Answer

'Error'

Correct!

Error

Correct Answer

error

Correct Answer

'error'

Correct Answer

"error"

Correct Answer

ERROR

Answer 5:

Correct! 0.5

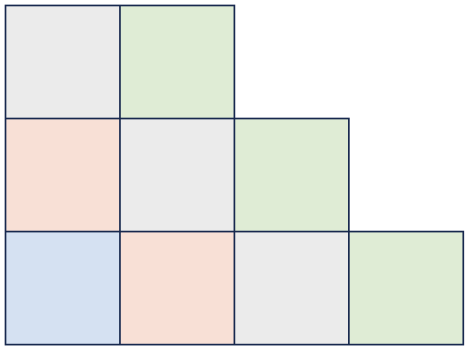
Correct!

0.5

Question 13

20 / 20 pts

Write a function **build( )** which takes two positive integers: the total number of blocks available (**t**) and the number of blocks that should be used for the first layer (**e**). The function returns a tuple containing two numbers: the number of layers in the built structure (**l**), and the number of leftover blocks (**n**). The first layer of the built structure must contain **e** blocks, and each subsequent layer must contain one less block than the layer below it. So, if you set t=15 blocks and need to have e=5 blocks for the first layer, you would end up with 5 layers and 0 leftover blocks.



```
def build(t, e):
    """Return the number of layers and leftover blocks
    >>> build(2,8) # not enough blocks to start the first layer
    (0, 2)

    >>> build(10,3) # 3 layers (3 + 2 + 1 = 6) + 4 unused blocks
    (3,4)

    >>> build(15,5) # 5 layers (5 + 4 + 3 + 2 + 1 = 15) + 0 unused blocks
    (3,4)

    We set two counters necessary to maintain the total number of blocks used in the structure (total_blocks_used)
    and to compute the number of blocks required for the subsequent layer (next_blocks)
    """
    total_blocks_used = _ _ _ _ _ (a) _ _ _ _ _
    next_blocks = _ _ _ _ _ (b) _ _ _ _ _
    _ _ _ _ _ (c) _ _ _ _ _
    _ _ _ _ _ (d) _ _ _ _ _

    # Keep building layers as long as there are enough blocks
    while t > total_blocks_used and next_blocks > 0:
        n -= _ _ _ _ _ (e) _ _ _ _ _
        # logical check to ensure the remaining number of blocks is never negative
        if n < 0:
            n += next_blocks
            break
        else:
            n = n
        _ _ _ _ _ (f) _ _ _ _ _   # Update the layer count
        _ _ _ _ _ (g) _ _ _ _ _   # Next layer has one less block
        _ _ _ _ _ (h) _ _ _ _ _   # Update the total blocks used counter

    # After loop, the remaining blocks will be in 'n' and 'l' is the number of layers built
    return (l,n)
```

a) [Select]

b) [Select]

c) [Select]

d) [Select]

e) [Select]

f) [Select]

g) [Select]

h) [Select]

Answer 1:

e

l

Correct!

0

= 0

n

= t

t

= e

Answer 2:

= t

= 0

Correct!

e

t

n

0

= n

= e

Answer 3:

l = 1

l != 1

Correct!

l = 0

l == 0

l == 1

l != 0

Answer 4:

n == t

n == e

n == next_blocks

n = e

n = next_blocks

n = 0

Correct!

n = t

n == 0

Answer 5:

0

e

t

total_blocks_used

Correct!

next_blocks

1

Answer 6:

l == 1

l += next_blocks

l -= 1

l -= next_blocks

l += n

Correct!

l +=1

Answer 7:

Correct!

next_blocks -= 1

next_blocks = next_blocks + 1

next_blocks = n + total_blocks_used

next_blocks = t - total_blocks_used

next_blocks = n

next_blocks += 1

Answer 8:

total_blocks_used += 1

total_blocks_used -= next_blocks

Correct!

total_blocks_used += next_blocks

total_blocks_used = n - next_blocks

total_blocks_used += n

total_blocks_used -= 1

Question 14

Original Score: 2.5 / 2.5 pts Regraded Score: 2.5 / 2.5 pts

This question has been regraded.

Which of the following code generate the same output? Choose **ALL** that apply.

A.

```
level = (1.4, 1.7, 1.5, 1, 0.7, 0.5)
for j in level:
    print(j)
```

B.

```
level = (1.4, 1.7, 1.5, 1, 0.7, 0.5)
for i in range(len(level)):
    print(level[i])
```

C.

```
level = (1.4, 1.7, 1.5, 1, 0.7, 0.5)
for k in range(len(level)):
    print(k)
```

D.

```
level = (1.4, 1.7, 1.5, 1, 0.7, 0.5)
m = 0
while m < len(level):
    print(level[m])
```

E.

```
level = (1.4, 1.7, 1.5, 1, 0.7, 0.5)
n = 0
while n < len(level):
    print(level[n])
    n += 1
```

F.

```
level = (1.4, 1.7, 1.5, 1, 0.7, 0.5)
p = 0
while p < len(level):
    print(p)
    p += 1
```

Correct!

- (●) A, B and E
- ( ) A, C and E
- ( ) A and B
- ( ) B and E
- ( ) C and F
- ( ) D, E and F
- ( ) A, B and D
- ( ) None of the options

Question 15

2.5 / 2.5 pts

Which of the following code generate the same output? Choose **ALL** that apply.

A.

```
data = [139, 152, 147, 30, 157, 187]
for i in data:
    if i % 3 == 0:
        break
    print(i)
print(i)
```

B.

```
data = [139, 152, 147, 30, 157, 187]
for j in data:
    if j >= 150:
        break
    print(j)
```

C.

```
data = [139, 152, 147, 30, 157, 187]
for k in data:
    if not k < 140:
        break
    print(k)
```

D.

```
data = [139, 152, 147, 30, 157, 187]
for m in data:
    if m >= 150:
        continue
    print(m)
```

Correct!

- (●) B and C
- ( ) A, B, C and D
- ( ) C
- ( ) B and D
- ( ) C and D
- ( ) D
- ( ) A, B, and C
- ( ) A and D
- ( ) A and B
- ( ) A
- ( ) B

Question 16
2.5 / 2.5 pts

Which of the following function definitions can be interchangeably used because they generate the same output? Choose **ALL** that apply.

A.

```
def func(r):
    return r + 3, r ** 2
```

B.

```
def func(r):
    return r + 3 + r ** 2
```

C.

```
def func(r):
    return r ** 2
    return r + 3
```

D.

```
def func(r):
    return r ** 2, r + 3
```

E.

```
func = lambda r : r + 3
```

F.

None of the options

- A and C

Correct!

- F
- A and C
- A and D
- D and E
- B and C
- B and E
- A, C and D

Question 17

2.5 / 2.5 pts

Which of the following function definitions can be interchangeably used because they generate the same output? Choose **ALL** that apply.

A.

```
def func(r):
    return r + 3, r ** 2
```

B.

```
def func(r):
    return r + 3 + r ** 2
```

C.

```
def func(r):
    return r ** 2
    return r + 3
```

D.

```
def func(r):
    return r ** 2, r + 3
```

E.

```
func = lambda r : r ** 2
```

F.

None of the options

- A and D
- F
- A and C
- C and D
- B and E
- B and C
- D and E

Correct!

- C and E

Question 18

8 / 16 pts

Write a function **func( )** that takes one string input and returns the number of occurrences of repeated consecutive characters. Whether the character is in uppercase or lowercase should not matter, i.e. consecutive "a" and "A" should be counted as a repeated occurrence. If the input word is not of type string, the function should return an error ("Error"). Examples are provided below. Use in-built python commands of type(), .upper() and .lower() that you previously used in your lab assignments to construct your function. Do not add any spacing before or after the round and square brackets, i.e. type(word), for example, should be written without any spaces.

```
def func(word):
    """ Return the number of occurrences of repeated consecutive characters in a string
    >>> func("aAb") # 1 repetition: a as A
    >>> func(777) # Error
    >>> func("three") 1 repetition
    >>> func("threee") 2 repetitions
    """
    if _ _ _ _ (a) _ _ _ _:
        _ _ _ _ (b) _ _ _ _
    i = _ _ _ _ (c) _ _ _ _
    for j in _ _ _ _ (d) _ _ _ _:
        if _ _ _ _ (e) _ _ _ _:
            i += _ _ _ _ (f) _ _ _ _
    return i
```

a) type(word)!=str

b) return "Error"

c) 0

d) range(len(word)-1)

e) word.lower()[j]==wo

f) 1

Answer 1:

Correct! type(word)!=str

Correct Answer

type(word) != str

Correct Answer

type(word) != str

Correct Answer

not type(word) == str

Correct Answer

not type(word) == str

Correct Answer

not type(word)==str

Correct!

type(word)!=str

Answer 2:

You Answered return "Error"

Correct Answer

print("Error")

Correct Answer

print('Error')

Correct Answer

print("error")

Correct Answer

print('error')

Correct Answer

print("Error")

Correct Answer

print("error")

Correct Answer

print('Error')

Correct Answer

print('error')

Answer 3:

Correct! 0

Correct!

0

Answer 4:

You Answered range(len(word)-1)

Correct Answer

range(len(word))

Correct Answer

range(len(word))

Correct Answer

range(len(word))

Answer 5:

You Answered word.lower()[j]==word.lower()[j+1]

Correct Answer

word[j].lower() == word[j-1].lower()

Correct Answer

word[j].lower()==word[j-1].lower()

Correct Answer

word[j].upper() == word[j-1].upper()

Correct Answer

word[j].upper()==word[j-1].upper()

Answer 6:

Correct! 1

Correct!

1

Question 19

9 / 9 pts

You have been tasked to write a program that controls the air conditioning in your classroom. The code below predicts the time required to bring the temperature of the room to the set temperature. Assume the code has been executed and answer the following questions.

```
def func(t, d=0, f=1):
     print(t) if t % 2 == 0 else None
     t = t - d - d * f
     return t

temp = 70
time = 1
while temp > 66:
    t = temp - 8.0
    func(temp - 2)
    temp, time = temp - 2, time * 2
    s=t
print("Set Temperature Attained!")
```

a) What is the first line of the output that gets displayed upon executing the code? 68

b) What is the final value associated with the variable **temp**? 66

c) What is the final value associated with the variable **time**? 4

Answer 1:

Correct! 68

Correct!

68

Correct Answer

68.0

Answer 2:

Correct! 66

Correct!

66

Correct Answer

66.0

Answer 3:

Correct! 4

Correct!

4

Correct Answer

4.0

Quiz Score: 100 out of 100

This quiz score has been manually adjusted by +8.0 points.