

My grades for CSC148_midterm1

Q1

0.33



01238022-WF80-4824-AC2C-0184ED408477
csc148_midterm1-1000f
#791 Page 2 of 14

1. (1 pt) General Concepts

Fill in the blanks with 3 different choices from the options given below, to correctly complete this sentence:
The interface of a class includes _____, _____ and _____.

- ☐ A. private attributes, if used in the implementation of public methods.
- ☒ B. representation invariants applicable to public attributes
- ☐ C. `__init__` & `__repr__` methods
- ☐ D. preconditions of public methods

One correct answer. B, 0.33

C, D are correct.

Q2

0

Circle ALL statements that apply.


- ☒ A. Every object has a type.
- ☐ B. Objects describe attributes, abilities and computational efficiency.
- ☐ C. Private methods and attributes in Python are not accessible to

The only correct answer is A. -1

C is incorrect: objects are instances of classes, and classes are 'blueprints' to create objects.

Q3 1

7921875c-7059-4703-8038-031638828795
0a0148_p010level-44082
#791 Page 3 of 14



3. (1 pt) General Concepts

PROGRAM:

```
1: def blah(lst: List[int]) -> Optional[Dict[int, int]]:  
2:     # You do not know anything about the internal  
                                     implementation of blah  
3:  
4: if __name__ == "__main__":  
5:     a = [1, 2, 3]  
6:     foo = blah(a)  
7:     print(a)  
8:     if 42 in foo:  
9:         print("Yes")  
10:  
11:     print("No")
```

Select ONE correct answer from below:

A. line 7 is guaranteed to print [1, 2, 3]
B. line 7 is guaranteed to print [1, 2, 3]
C. line 7 may or may not print [1, 2, 3]
D. an exception being raised.
E. true and find 42 in foo.
F. false and never find 42 in foo.
G. Exception being raised.

II. All of the above.
I. None of the above.

Good job! 1

3

Q4

0



AA25C54D-4753-4252-878C-4B5C9FA086A
cwi14f_n00taw0-9046f
#791 Page 4 of 14

4. (1 pt) Memory model
Consider the following interaction with the python console:

```
>>> lst = [1, 4, 8]  
>>> id(lst)  
8839458  
>>> lst.append("widtara")  
>>> id(lst)  
8839458
```

Which of the statements **best** describes what has occurred in the python console? **Select ONE correct option from below.**

- ☒ A. The python terminal shows that when we concatenate two lists, the memory address of the object will stay the same.
- ☐ B. The python terminal shows that when we use the id function, it will always stay the same.
- ☐ C. The python terminal shows that when we use a list method like `lst.append()` as a result, Python keeps the list as a new object. When we append a list to include a new element, it is not added, which results in Python keeping the same memory address.
- ☐ E. All of the above.
- ☐ F. None of the above.


Incorrect. -1

Q5 1

3E792F68-7784-454F-9081-38E19312F62A

040148ac08eac0-0a0d4f

8791 Page 5 of 14



5. (1 pt) Testing

A student asks, "Why do we bother writing test cases using the pytest module when we can just write doctests?" Select ONE correct option from below.

A. The pytest module automatically chooses good test cases for us by creating meaningful categories.

☒ B. The pytest module allows us to unit test functions in a separate file without compromising the readability of the code.

☐ C. The pytest module doesn't exist in Python, so we must always module do property-based testing, which generation inputs.

☐ D. The pytest module is used when we have two or more python should use doctests.

☐ E. The pytest module makes it easier to create sample function calls, which can help a programmer write client code.

☐ F. The pytest module improves the plasticity of the code by using assert statements.

☒ G. All of the above.

☐ H. None of the above.

6. (1 pt) ADTs

Which of the following statement(s) is/are true.

Circle ALL statements that apply.

A. Stacks are unique to the Python programming language

☒ B. Stacks follow the FIFO (First in First out) principle.

☐ C. Stacks follow the LIFO (Last in First out) principle.

☒ D. Stacks can be implemented with built in Python lists.

☐ E. Common methods for stack interfaces are: insert, remove, delete, empty, size

☒ F. Stacks can be implemented using queues.

☒ G. Queues can be implemented using stacks.

☐ H. All of the above.


☐ I. None of the above.

5

Three -0.5
cor-
rect
an-
swers.
C, D,
F, and
G are
cor-
rect.

Q7

2.5



03A35226-6960-4408-9EAE-16E23C7DCDC9
csc141_410000-10000
#791 Page 6 of 14

7. (4 pts) Memory model diagram

Here is a short python program:

```
1 def do_something(a: int, b: List[int]) -> None:
2     l = b
3     l.append(a)
4     a += a * 2
5
6
7 if __name__ == '__main__':
8     nns = 100
9     lnt = [1]
10    do_something(nns, lnt)
```

a. (3 pts) Trace the python code above and draw the memory model diagram to show the state of the program after all lines (1 - 10) in the program have been executed. **Hint:** Please remember to use unique ids for any object you create, and don't forget to remove frames by **lightly crossing it out** from the call stack once a function has finished executing. **Show all your work!**

do_something

id3

0

id2

id4

list

id2

int

8. (1 pt) Briefly provide **one** reason as to why memory model diagrams are useful.

It is helpful to trace the flow of the code and visualize the allocation of variables

b should be in the stack frame. -0.5

Q8

1

Great job! Answer 1 refers to tracing and/or debugging.

Q9a

2

7831004-0571-4270-8208-62565A20014
unc148_pilfocal-hu4f
8791 Page 7 of 14

9. (6 pts) Inheritance and Abstraction

a. (2 pts) For each pair of classes, check the box to indicate whether they have a Composition relationship, an Inheritance relationship, or neither.

Class 1	Class 2	Composition?	Inheritance?	Neither?
VendingMachine	Product	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Product	Chips	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Drink	Product	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Coin	VendingMachine	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Q9b

2

b. (2 pts) From the five classes introduced in the table, VendingMachine, Product, Chips, Drink, and Coin, which one of the classes should be made as an abstract class? What is the purpose of creating this abstract class?

The Product class must be an abstract class, because product is category which means the product is just category where chips and drink belongs to, the purpose of abstract class is to get the desired

Q9c

1

c. (1 pt) What is one possible representation invariant for the Coin class? *For any similar characteristics*

d. (1 pt) Consider the following Python code. You can assume the code works without errors.

```
products = List[Product]
for pro in products:
    pro.get_price()
```

The code shows that we easily loop through the products even though the vending machine has different types of products (i.e., Chips and Drink). What concept from the course does this piece of code illustrate?

Q9d

0.5

According to the concept of inheritance, the type of child class can be same as their parent class, since the child class has all of the attributes and methods from their parent class

partial grade 0.5

for 'inheritance' or 'abstraction'.

Correct answer is Polymorphism



25930763-9536-4605-8766-7E1A2DFED3A5
 cu:48_4d0e0e0-760f
 #791 Page 8 of 14

10. (2 pts) Stacks

Given the Stack implementation, in the space below write a `reverse2` method which reverses the order of the bottom two elements of a stack. If the stack size is less than 2, the method has no effect.

PROGRAM:

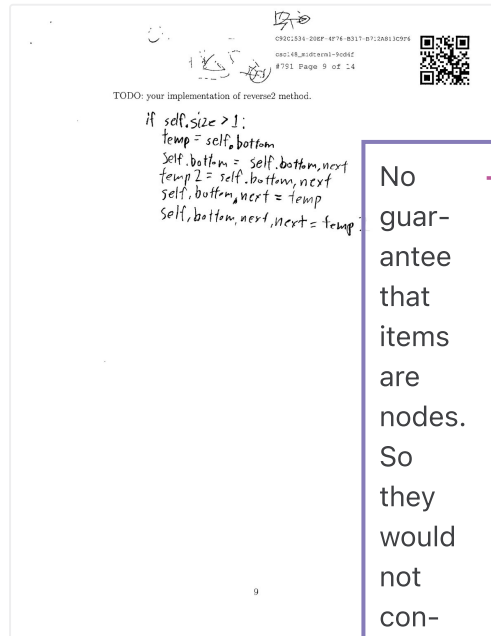
```
1 class Stack:
2     def __init__(self, item):
3         self.item = item
4         self.size = 0
5
6     def push(self, item):
7         self.item = item
8         self.size += 1
9         self.top = item
10        self.size = 0
11
12
13
14    def reverse2(self):
15        """Reverses the order of the bottom two elements of a
16        stack.
17
18
19        """
20        >>> s = Stack()
21        >>> s.push(1); s.push(2); s.push(3); s.push(4);
22        >>> s
23        1, 2, 3, 4
24        >>> s.reverse2()
25        >>> s
26        2, 1, 3, 4
27        """
```

Write the implementation (aka code) for `reverse2` on the NEXT page.

8

Q10

0.5



No guarantee that items are nodes. So they would not contain a "next" attribute, or a different "next" attribute. See docstring, for example Use stack methods instead

-1.5



45C4P5D5-0L76-44P7-81D3-03E41181D3D5
001148_000000-000000
#791 Page 10 of 14

11. (2 pts) Stacks and Queues output

Given the stack and queue implementations from prep4, in the space below, what is the output of this program?
By default print prints a new line, the end parameter allows us to replace the newline with another character. Essentially, line 12 prints s; and line 16 prints q. If there is an error or exception indicate on which line it occurs.

PROGRAM:

```
1 s = Stack()
2 q = Queue()
3 s.push(5)
4 q.enqueue(2)
5 s.push(3)
6 q.enqueue(9)
7 s.push(1)
8 q.enqueue(7)
9 s.push(q.dequeue())
10 q.enqueue(s.pop())
11 q.enqueue(q.dequeue())
12 print("s", end=" ")
13 while not s.is_empty():
14     print(s.pop(), end=" ")
15 print()
16 print("q", end=" ")
17 while not q.is_empty():
18     print(q.dequeue(), end=" ")
```

s: 5, 3, 1, 7
q: 2, 9

TODO: What is the output of this program?

s: 1 3 5
q: 7 2 9

Q11

2

correct output, good job! 2

Q12

3.5

HEADID=761a-004f-0000-000000000000
GUESTID=guest-14484
#391 Page 11 of 14

12. (4 pts) Queues

Implement a function that removes all even integer values from a queue of integers. The integers in the queue should be smaller than or equal to a given minimum.

```
def even_min_filter(q: Queue[int], minimum: int) -> None:
    """Remove all even integers from <q> that are smaller than
    or equal than <minimum>."""

    >>> q = Queue()
    >>> q.enqueue(4)
    >>> q.enqueue(2)
    >>> q.enqueue(7)
    >>> q.enqueue(9)
    >>> even_min_filter(q, 6)
    >>> q.is_empty()
    >>> False
    >>> q
    7 9
    >>>
```

TODO: Implement the function even_min_filter below

```
temp = Queue()
while not q.is_empty():
```

+2 for putting the cond
correctly

+1 for de-
queue-
ing
ing back
into the
queue

1

1

Q13

1.5



20240209-19020-4952-8750-52A1343C83A
encl04_address-1048f
#791 Page 12 of 14

13. (2 pt) Linked lists

In the space below, briefly (2-3 sentences) explain why inserting an item at index 0 is faster when using a linked list implementation than when using Python's built-in list implementation.

-0.5: Did not mention about the list has to shift n items in list. 1.5

14. (7 pts) Linked lists

Implement the following function according to its description. For this question, you should refer to the documentation of the `LinkedList` class found on the old

Ans: it's faster because inserting at position 0 with `LinkedList`, as opposed to array-based, does not have to move n items to make room, instead it adjusts the `_first` reference.

present the `cycled` function on the NEXT page!

12


Q14

4

picked 3
up every
index ob-
ject
correctly!

bounds

with
mistakes



804800C3-687D-4687-8472-86878615C889

cm1186_nintend-70d46f

8791 Page 14 of 14

[This page is for scratch work and will not be graded unless specified.]

14

