CMPT 120-D400 Midterm Exam Sample

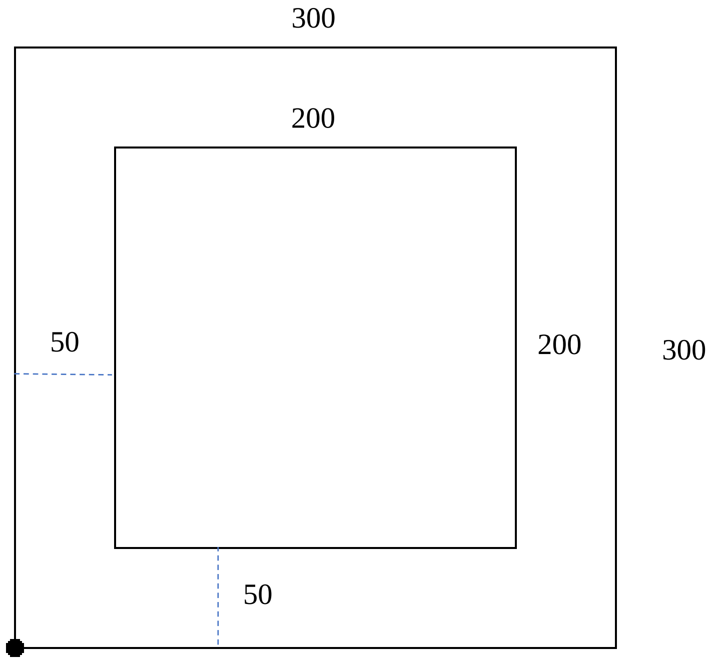
Fall 2022

Coding Questions

This is a **60 minute closed book exam**: notes, books, computers, calculators, electronic devices, etc. are **not** permitted. Do not speak to any other students during their exam or look at their work. Please remain seated and **raise your hand** if you have a question.

Turtle Graphics

(10 marks) Write a **complete** Python turtle graphics program that draws this picture:



- Assume the turtle starts at the black dot in the lower left corner facing east (i.e. to the right).
- The dimensions of the outer square are 300 by 300, and the dimensions of the inner square are 200 by 200.
- As shown by the dashed line, there is a 50 pixel gap all around.
- The only module you should import is `turtle`. You don't need to write any of your own functions, but can if you like.
- Just draw the two squares. **Don't** draw the dot, numbers, or dashed lines.

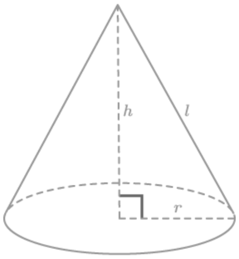
Solution

```
import turtle  # import: 2 marks

turtle.forward(300)   # outer square: 2 marks right shape; 1 mark right size
turtle.left(90)
turtle.forward(300)
turtle.left(90)
turtle.forward(300)
turtle.left(90)
turtle.forward(300)

turtle.up()           # re-position turtle: 2 marks
turtle.left(90)
turtle.forward(50)
turtle.left(90)
turtle.forward(50)
turtle.right(90)
turtle.down()

turtle.forward(200)   # inner square: 2 marks right shape; 1 mark right size
turtle.left(90)
turtle.forward(200)
turtle.left(90)
turtle.forward(200)
turtle.left(90)
turtle.forward(200)
```

Functions

a) (5 marks) The volume of a **cone** is given by this formula:

$$\frac{1}{3}\pi r^2 h$$



h is the height of the cone, and *r* is the radius of the bottom circle.
The variable *l* in the diagram is not used in the formula.

Write a Python function called `cone_volume(r, h)` that uses this formula to calculate and return the volume of a cone with the given radius and height. If `r` or `h` is 0, or less, then the function should return 0.

Use 3.14 for the value of π.

Here's how it should work:

```
>>> cone_volume(2, 10.8)
45.21600000000001
>>> cone_volume(-2, 10)
0
```

Solution

```
def cone_volume(r, h):
    if r <= 0 or h <= 0: return 0
    return 3.14 * r * r * h / 3
```

b) (5 marks) Write a Python function called `describe(country, capital)` that returns a string saying that `capital` is the capital city of `country`. Format the string exactly as in these examples:

```
>>> describe('Canada', 'Ottawa')
"Ottawa is Canada's capital city"

>>> describe('Australia', 'Canberra')
"Canberra is Australia's capital city"
```

Solution

```
def describe(country, capital):
    return capital + ' is ' + country + "'s capital city"
```

Selection

(5 marks) Write a **complete** Python program that asks the user enter two numbers, x and y, and then prints:

- "x and y are the same" if x and y are equal; replace x and y in the string with their values (as shown below)
- "a is less than b", where is a is the min of x and y, and b is the max of x and y (as shown below)

Important *Don't* use the built-in Python `min` and `max` functions in your program, and *don't* import any modules.

Assume the user enters valid `float`s.

Here's a sample run:

```
What is x? 4.2
What is y? -7
-7.0 is less than 4.2
```

And another:

```
What is x? 3.1
What is y? 3.1
3.1 and 3.1 are the same
```

Solution

```
#
# Other forms if-elif statements are possible
#

# 1 mark for consistent and correct indentation
#
x = float(input('What is x? '))        # 1 mark for reading x and y
y = float(input('What is y? '))        #        as floats
if x == y:                             # 1 mark for x == y case
    print(f'{x} and {y} are the same')
elif x < y:                            # 1 mark for x < y case
    print(f'{x} is less than {y}')
else:                                  # 1 mark for y < x case
    print(f'{y} is less than {x}')  # note x, y swapped
```

Iteration

a) (5 marks) Write a complete Python program that uses a **while-loop** (and no other kind of loop!) to print the multiples of 5 from 1 to 1000:

```
5
10
15
20
…
995
1000
```

Solution

```
n = 5                   # initialization
while n <= 1000:        # while-loop header
    print(n)            # print
    n += 5              # increment
```

b) (5 marks) Write a complete program that asks the user to enter a string. If they **don't** enter "yes" or "no" then it asks them to try again (as shown in the sample run). It keeps re-asking them until they enter "yes" or "no", and finally prints "Got it!" when they do

Here are three sample runs:

```
Yes or no? yep
Sorry, I don't understand that.

Yes or no? nope
Sorry, I don't understand that.

Yes or no? yes
Got it!
```

Solution

```
ans = input('Yes or no? ')                         # get input
while not (ans == 'yes' or ans == 'no'):           # header/condition
    print('Sorry, I don\'t understand that.') # body
    ans = input('Yes or no? ')
print('Got it!')                                   # final message
```