10.009 The Digital World

Term 3. 2020

Problem Set 2 (for Week 2)

Last update: January 31, 2020

Due dates:

- **Problems: Cohort sessions**: Following week: Tuesday 11:59pm.
- **Problems: Homework**: Same as for the cohort session problems.
- **Problems: Exercises**: These are practice problems and will not be graded. You are encouraged to solve these to enhance your programming skills. Being able to solve these problems will likely help you prepare for the midterm examination.

Objectives:

1. Learn to write and use functions.
2. Learn basic techniques for program debugging.
3. Learn to input data from the keyboard and handle user input.
4. Learn to use if-elif-else statement.

Note: Solve the programming problems listed below using your favourite text editor. Make sure you save your programs in files with suitably chosen names, **and try as much as possible to write your code with good style (see the style guide for python code)**. In each problem find out a way to test the correctness of your program. After writing each program, test it, debug it if the program is incorrect, correct it, and repeat this process until you have a fully working program. Show your working program to one of the cohort instructors.

Problems: Cohort sessions

1. *Functions: multivalued:* Recall the following equation that gives the height of a ball at time t when it is thrown with an initial velocity of v_0.

   $$y(t) = v_0 t - \frac{1}{2} g t^2$$

   Write a function named `position_velocity()` that takes v_0 and t as inputs and returns $y(t)$ and $y'(t)$, where $y'(t)$ is the first derivative of $y(t)$ with respect to t. Use your knowledge of calculus to find the formula to calculate $y'(t)$. Define the gravitational constant g in your code. **Round the output to two decimal places** using `round(n,d)`. Use $g = 9.81\ m\ s^{-2}$.

   ```
   >>>print(position_velocity(5.0, 10.0))
   (-440.5, -93.1)
   >>>print(position_velocity(5.0, 0.0))
   (0.0, 5.0)
   >>>print(position_velocity(0.0, 5.0))
   (-122.62, -49.05)
   ```

2. *Functions: BMI:* Write a function that takes in your weight in kilograms and your height in centimetres, and returns your Body Mass Index (BMI). The BMI can be calculated by taking the weight in kilograms and dividing it by the square of your height in meters. You can assume that the height is never zero. Note that one meter is one hundred centimeter. Round the output to one decimal place using `round()` function. For example:

   ```
   >>>print(bmi(60,120))
   41.7
   >>>print(bmi(50,150))
   22.2
   >>>print(bmi(43.5,142.3))
   21.5
   ```

   Write a test program that prompts the user to enter the weight and height as follows.

   ```
   Weight in kilograms: 95.5
   Height in centimetres: 180
   BMI: 29.5
   ```

   Vocareum Submission: Submit only the function definition without the test program.

3. *Functions: Area of a Triangle:* Write a function that takes in the three points (x1,y1), (x2,y2), and (x3,y3) of a triangle and returns its area. The formula for computing the area

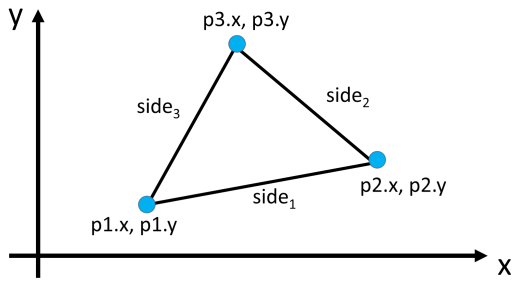
of a triangle is

$$\begin{aligned} s &= (side_1 + side_2 + side_3)/2 \\ area &= \sqrt{s(s-side_1)(s-side_2)(s-side_3)} \end{aligned}$$

where $side_n$ is the length of one of the sides of a triangle. The three points are of the type Coordinate defined as:

```
class Coordinate:
  x = 0.0
  y = 0.0
```

See the figure below. **Round your answer to two decimal places**. To test the function,



you can use the following code:

```
print("Test Case 1")
p1=Coordinate()
p1.x=1.5
p1.y=-3.4
p2=Coordinate()
p2.x=4.6
p2.y=5
p3=Coordinate()
p3.x=9.5
p3.y=-3.4

ans=area_of_triangle(p1,p2,p3)
print(ans)

print("Test Case 2")
p1=Coordinate()
p1.x=2.0
p1.y=-3.4
p2=Coordinate()
p2.x=4.6
p2.y=5
p3=Coordinate()
p3.x=9.5
p3.y=-1.4

ans=area_of_triangle(p1,p2,p3)
print(ans)

print("Test Case 3")
```

```
p1=Coordinate()
p1.x=1.5
p1.y=3.4
p2=Coordinate()
p2.x=4.6
p2.y=5
p3=Coordinate()
p3.x=-1.5
p3.y=3.4

ans=area_of_triangle(p1,p2,p3)
print(ans)

print("Test Case 4")
p1=Coordinate()
p1.x=-1.5
p1.y=3.4
p2=Coordinate()
p2.x=4.6
p2.y=5
p3=Coordinate()
p3.x=4.3
p3.y=-3.4

ans=area_of_triangle(p1,p2,p3)
print(ans)
```

The expected output should be:

```
Test Case 1:
33.6
Test Case 2:
28.9
Test Case 3:
2.4
Test Case 4:
25.38
```

Write a test program that prompts the user to enter the coordinates of the three sides and display the area. Here is a sample run:

```
Enter x coordinate of the first point of a triangle: 1.5
Enter y coordinate of the first point of a triangle: -3.4
Enter x coordinate of the second point of a triangle: 4.6
Enter y coordinate of the second point of a triangle: 5
Enter x coordinate of the third point of a triangle: 9.5
Enter y coordinate of the third point of a triangle: -3.4
The area of the triangle is 33.6
```

Vocareum Submission: Submit only the function definition without the test program.

4. *Program tracing:* What is printed when the following Python code snippets are executed? For each problem assume three cases: Case 1: $x < y$, Case 2: $x > y$, and Case 3: $x = y$.

   Note: Submit the answer on eDimension.

   (a)
   ```
   if x<y:
       print(x)
   elif x>y:
       print(y)
   else:
       print(x+y)
   ```

   (b)
   ```
   if x<y:
       print(x+y)
   elif x>y:
       print(y)
       if x==y:
           print(x-y)
   else:
       print(x)
   ```

5. *Functions and conditions:* Write a function called `describe_bmi` that takes in a BMI value, and returns a string. The output string should follow the conditions specified by the Healthhub.sg (`https://www.healthhub.sg/live-healthy/410/Healthy%20Weight`).

   - $bmi < 18.5$: nutritional deficiency
   - $18.5 <= bmi < 23$: low risk
   - $23 <= bmi < 27.5$: moderate risk
   - $27.5 <= bmi$: high risk

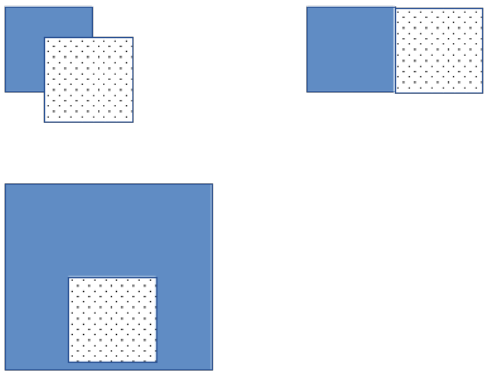
   An example of test cases are:

   ```
   >>> describe_bmi(18)
   nutritional deficiency
   >>> describe_bmi(18.5)
   low risk
   >>> describe_bmi(20)
   low risk
   >>> describe_bmi(23)
   moderate risk
   >>> describe_bmi(27.5)
   high risk
   >>> describe_bmi(30)
   high risk
   ```

6. *Functions and conditions:* Write a function named `letter_grade` that takes an integer argument, which represents a *mark* of a student. Return 'A' if $mark \geq 90$, 'B' if $80 \leq mark < 90$, 'C' if $70 \leq mark < 80$, 'D' if $60 \leq mark < 70$, and 'E' if $mark < 60$. Return `None` if it is not a valid mark. A valid mark ranges from 0 to 100.

```
>>>print(letter_grade(102))
None

>>>print(letter_grade(100))
A

>>>print(letter_grade(83))
B

>>>print(letter_grade(75))
C

>>>print(letter_grade(67))
D

>>>print(letter_grade(52))
E

>>>print(letter_grade(-2))
None
```

7. *Functions and Conditionals: Two Squares:* Write a function called `is_in_square()` that takes in two `Coordinate` objects and two floats that represent two squares. Each square is represented by a `Coordinate` that specifies its centre as well as a float that specifies its side. If the second square is within the first square, or overlaps the first square, the function should return `True`, otherwise it should return `False`. The diagram shows all the cases where the function should return `True`. In the case on the right, it is `True` because the edges of the two squares overlap.



Design your own test cases and verify the function. An example would be:

```
>>> s1 = Coordinate()
>>> s1.x, s1.y = 10, 10
>>> s2 = Coordinate()
>>> s2.x, s2.y = 20, 10
>>> is_in_square(s1, 5, s2, 4)
False
```

Problems: Homework

1. *Functions: Temperature conversion:* Write a function named `fahrenheit_to_celsius`. This function takes a fahrenheit value as input and returns its centigrade equivalent. Recall: If C denotes the centigrade value and F its fahrenheit equivalent, then $F = C * 9/5 + 32$. For example:

```
>>>print(fahrenheit_to_celsius(32))
0.0
>>>print(fahrenheit_to_celsius(-40))
-40.0
>>>print(fahrenheit_to_celsius(212))
100.0
```

2. *Functions: Temperature conversion:* Write a function named `celsius_to_fahrenheit` that returns the fahrenheit equivalent of a centigrade value input as an argument.

```
>>>print(celsius_to_fahrenheit(0))
32.0
>>>print(celsius_to_fahrenheit(-40))
-40.0
>>>print(celsius_to_fahrenheit(100))
212.0
```

3. *Functions: Volume of a cylinder* Write a function that takes in the radius and the length of a cylinder and returns the area and volume using the following formulas:

$$\begin{aligned} area &= radius * radius * \pi \\ volume &= area * length \end{aligned}$$

Round your answer to two decimal place and output both the area and volume as a tuple.

For example,

```
>>>print(area_vol_cylinder(1.0,2.0))
(3.14, 6.28)

>>>print(area_vol_cylinder(2.0,2.3))
(12.57, 28.9)

>>>print(area_vol_cylinder(1.5,4))
(7.07, 28.27)

>>>print(area_vol_cylinder(2.2,5.0))
(15.21, 76.03)
```

4. *Functions: Wind-chill temperature* How cold is it outside? The temperature alone is not enough to provide the answer. Other factors such as wind speed, relative humidity, and sunshine play important roles in determining coldness outside. In 2001, the National

Weather Service (NWS) implemented the new wind-chill temperature to measure the coldness using temperature and wind speed.

$$t_{wc} = 35.74 + 0.6215t_a - 35.75v^{0.16} + 0.4275t_a v^{0.16}$$

where t_a is the outside temperature measured in Fahrenheit and v is the speed measured in miles per hour. t_{wc} is the wind-chill temperature.

Write a function that takes in a temperature and a wind speed and returns the wind-chill temperature.

```
>>>print(wind_chill_temp(5.3,6))
-5.56706845588

>>>print(wind_chill_temp(2.2,4))
-6.34646224199
```

Write a test program that prompt the user to key in the temperature and the wind speed as follows.

```
Outside temperature in Fahrenheit: 5.3
Wind speed in  miles per hour: 6
wind chill index: -5.56707
```

Vocareum Submission: Submit only the function definition without the test program.

5. *Functions: Number of years and days:* Write a function `minutes_to_years_days` that takes in minutes as its input parameter, and returns the number of years and days for the minutes. For example, if the function takes in 527040 minutes, it is equivalent to 366 days, and so it will return 1 year and 1 day. *For simplicity, assume a year has 365 days and round down the remaining days.* For example,

```
>>>print(minutes_to_years_days(1000000000))
(1902, 214)
>>>print(minutes_to_years_days(2000000000))
(3805, 63)
```

Test the function by writing a test program that prompts the user, using `input`, to enter the minutes and print the output of the function. Here is a sample run:

```
Enter the number of minutes: 1000000000
1000000000 minutes is approximately 1902 years and 214 days.
```

6. *Functions: Future investment value:* Write a function that takes in an investment amount, the annual interest rate in percentage term, and the number of years, and returns the future investment value using the following formula:

$$futureInvestmentValue = investmentAmount \times (1 + monthlyInterestRate)^{numberOfMonths}$$

For example:

```
>>>print(investment_val(1000,4.25,1))
1043.34
>>>print(investment_val(1500,3.25,2))
1600.6
>>>print(investment_val(1000,2.25,0.5))
1011.3
>>>print(investment_val(2000,4.25,3))
2271.46
```

Round the answer to two decimal places.

Write a test program that keys in the investment amount, annual interest rate, and the number of years. Here is a sample run:

```
Enter investment amount: 1000
Enter annual interest rate (%): 4.25
Enter number of years: 1
Accumulated value is 1043.33
```

Vocareum Submission: Submit only the function definition without the test program.

7. *Functions: largest number:* Write a function named `is_larger` that takes two numbers, say $n1$ and $n2$, as inputs. If $n1 > n2$ the function returns `True` otherwise it returns `False`. You can assume that the inputs are always numbers, i.e. either integers or floats.

```
>>>print(is_larger(2,-1))
True

>>>print(is_larger(-1,2))
False

>>>print(is_larger(2,2))
False
```

8. *Functions: Arguments:* Write a function named `check_value` that takes four integers $n1$, $n2$, $n3$, and x as inputs. The function returns `True` if x is greater than $n1$ and $n2$ but is less than $n3$, otherwise it returns `False`.

```
print("Test case 1: check_value(1,4,8,7)")
print("ans = True")
ans=check_value(1,4,8,7)
```

```
print(ans)

print("Test case 2: check_value(10,4,8,7)")
print("ans = False")
ans=check_value(10,4,8,7)
print(ans)

print("Test case 3: check_value(1,10,8,7)")
print("ans = False")
ans=check_value(1,10,8,7)
print(ans)

print("Test case 4: check_value(1,4,5,7)")
print("ans = False")
ans=check_value(1,4,5,7)
print(ans)
```

The output should be:

```
Test case 1: check_value(1,4,8,7)
ans = True
True
Test case 2: check_value(10,4,8,7)
ans = False
False
Test case 3: check_value(1,10,8,7)
ans = False
False
Test case 4: check_value(1,4,5,7)
ans = False
False
```

End of Problem Set 2.