HW7 (100 POINTS)

These instructions are applicable for all scripts that you will write in python3.

```
Instructions for Python3 Homework submission
#class header

#Q1
var1=124

#Q2
var2=3+var1
```

```
This entire HW is not divided in parts, and it is due Wed March 13th
11:59pm.

You will get all 8 extra credit points for submitting all the scripts by
Wed 11:59pm.
```

Policy: Work on the HW problems on your own. It is not allowed to share code in anyway. The use of chatGPT is not allowed. Using internet to find out how to solve problems is not allowed.

General Grading rules we will follow:

- We do not give points for instructions but take off points for not following them.
- We take all points off for a question if your code has syntax errors (comment out the code for partial credit).
- We grade the code, not the output of the code. Even if the output is correct, hard-cording, redundant logic, code that has not purpose in the program will not earn full credits.
- We will take points off if you use out-of-class material.
- We will take points off if you do not follow directions when a question specifies to use a certain command or write the code in a certain way.

1. **(25)** The provided script **ex7-planets-weight.py** contains a dictionary D, where the keys are planet names, and the values are the gravitational constants on the corresponding planet. The gravitational constants are normalized to the gravity of the Earth.

   Here is the dictionary:

```
D={"venus":0.9,"mars":0.38,"mercury":0.38,"jupiter":2.36,"saturn":0.92,
"uranus":0.89,"neptune":1.13,"pluto":0.07}
```

Edit the script and add the following after the dictionary *D*.

Q1 (17) Make a function called ***weight_planet*** that takes **no parameters**. The function:

- Has a documentation string, describing the function's task.
- Prints a list type of the dictionary keys.
- Prompts the user to enter the name of one planet e.g.,
  Enter the name of one of the planets listed above.
  The user's input should be stored in a variable called *name.*
- Prompt the user to enter some weight in kg (weight should be a real number) and stores the user's input in a variable called *we.*
- Calculates the weight on the planet chosen by the user by accessing values in the dictionary. To calculate the weight on a planet, multiply the weight on earth*, we,* by the normalized gravitational constant on that planet (i.e., the value in the dictionary corresponding to the specified planet). Store the result in variable *wp.*
- Returns a tuple type containing both the name of the planet chosen by the user, i.e., *name,* and the weight on that planet, i.e., *wp.*

Q2 (5) Call the function and store what the function returns in a variable called *weight_name,* or alternatively in two separate variables *weight* and *name*.

Q3 (3) Print the name of the planet and the weight on that planet by using the variable(s) you made in Q2.

For example, for the planet venus and a weight of 50.0 kg the output should be:

```
Your weight on planet venus is 45.00 kg.
```

2. **(25)** All humans display four different types of electrical patterns or "brain waves" across the cortex. The brain waves can be observed with an EEG (or an "electroencephalograph") – a tool that allows researchers to note brain wave patterns. Each brain wave has a purpose and helps us in optimal mental functioning.

Brain waves have frequencies ranging roughly from 1.0 to 100.0 Hz (assume 1.0 and 100.0 are included). The five types are classified based on their frequency:

Gamma:	35 Hz	<	frequency	<= 100 Hz
Beta:	12 Hz	<	frequency	<= 35 Hz
Alpha:	8 Hz	<	frequency	<= 12 Hz
Theta:	4 Hz	<	frequency	<= 8 Hz
Delta	1 Hz	<=	frequency	<= 4 Hz

Make a script called **ex7-brainwaves.py** and in it include the following:

Q1. (18) Make a function called ***func_waves*** that takes one argument, which is one frequency (one real number). The function should print a statement reporting the frequency and the wave type:

For example:

if the frequency is 92.93 the function will print:

The frequency 92.9 HZ is Gamma type.

if the frequency is 6.433 the function will print:

The frequency 6.4 HZ is Theta type.

and so on.

Ignore the case in which the user inputs a frequency which is out of the range [1.0 -100.0] Hz (i.e., the function does not need to print anything in this case). The out of range should not be included anywhere in your if statement.

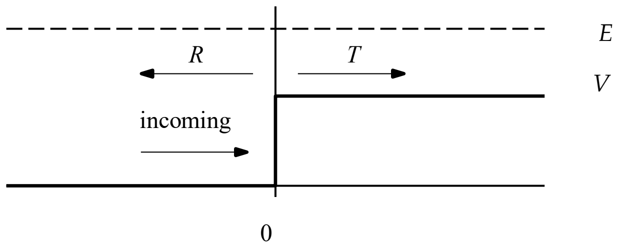
Q2. (7) Loop over elements of this list:

```
L=[99.33,30.21,10.0,5.0,2.0]
```

and for each element run the function ***func_waves***. In this way you will test all the 5 cases. The output should be:

```
The frequency 99.3 Hz is Gamma type.
The frequency 30.2 Hz is Beta type.
The frequency 10.0 Hz is Alpha type.
The frequency 5.0 Hz is Theta type.
The frequency 2.0 Hz is Delta type.
```

3. **(35)** A well-known quantum mechanics problem involves a particle of mass m that encounters a one-dimensional potential step, like this:



The particle has initial kinetic energy E and wavevector $k_1 = \frac{\sqrt{2mE}}{\hbar}$. It enters from the left (*incoming*) and encounters a sudden jump in potential energy of height V at position x = 0. $\hbar$ is the reduced Planck constant.
By solving the Schrödinger equation, one can show that when E > V, the particle may either (a) pass the step (*T*), in which case it has a lower kinetic energy of E−V on the other side and a correspondingly smaller wavevector of $k_2 = \frac{\sqrt{2m(E-V)}}{\hbar}$, or
(b) it may be reflected (*R*), keeping all its kinetic energy and having an unchanged wavevector but moving in the opposite direction.

By setting the reduced Planck's constant $\hbar = 1$, k_1 and k_2 become:

$k_1 = \sqrt{2mE}$ *Eq1*

$k_2 = \sqrt{2m(E-V)}$ *Eq2*

The probabilities T for transmission and R for reflection are given by:

$T = \frac{4\,k_1\,k_2}{(k_1+k_2)^2}$ $Eq3$

$R = \frac{(k_1-k_2)^2}{(k_1+k_2)^2}$ $Eq4$

You will write functions to implement the formulas reported above for calculating the transmission and reflection probabilities.

In a Python program called **ex7-Qpotential.py** do the following:

Q1 (2) Import the required function from the math module. **Use function import**.

Q2 (6) Write a function called **f_k1** that takes two parameters, m, and E, and returns k_1 according to Eq1.

Q3 (6) Write a function called **f_k2** that takes three parameters, m, E and V, and returns k_2 according to Eq2.

Q4 (12) Write a function called **T_R** that takes m, E and V, uses **f_k1** and **f_k2** for calculating both T and R (based on respectively Eq3 and Eq4), and returns both T and R.

Q5 (9) Call function **T_R** to calculate the probabilities of transmission and reflection for a particle with mass equal to the electron mass $m = 9.11 \times 10^{-31}$ kg and energy E=10 eV encountering a potential step of height V=9 eV.

Use scientific notation for the electron mass value.

Print this formatted statement to screen, reporting the results.

```
T = 0.73 and R = 0.27, and their sum is 1.00
```

Notice that you should also calculate the sum of T and R.
You can either use the variable storing the returned tuple from the function, or you can unpack that tuple, but do not use indexing to access single elements in that tuple.

```
Submit to Gradescope HW7:
ex7-planets-weight.py
ex7-brainwaves.py
ex7-Qpotential.py
```