Assign 4

Q1.
create a function f1 which accepts 2 positional arguments and 2 key word arguments (with default values)
which are all numbers and return the sum
create a function f2 which accepts 3 positional arguments and 2 key word arguments (with default values)
which are all numbers and return the sum and difference of the alternate numbers i.e. v-w+x-y+z
create a function f3 which accepts 2 keyword arguments (with default values) and returns a list of the length of the
second argument with values all equal to the first argument (Hint: use list * operator)
create a function f4 which accepts 1 keyword argument (with default value) and returns a dictionary whose keys are
alphabets (a,b,c...) and values are numbers from 0 to the n-1 where n is the parameter value so {'a',:0,'b':1 ...}
create a function f5 which accepts a list of 2 functions and unknown number of positional arguments. It takes the
list of functions and calls each of first function in the list and passes the positional arguments of f5 to it.
The value returned is passed to the second function in the list. The first function in the list are either f1 or f2
the second function in the list are either f3 or f4. Compare the results of execution of the following:
l1=[f1,f3]
f5(l1,2,3,4)
l2=[f2,f3]
f5(l2,2,3,4)
l3=[f1,f4]
f5(l3,2,3,4)
l4=[f2,f4]
f5(l4,2,3,4)

Q2.
1. define a function singleStringPrint to receive a single argument which needs to be printed - if the argument received
is a number (int or float) print using the .format style for float, if the argument is a string then print using the
.format style for string.
2. define a function singleNumericPrint to receive a single argument which needs to be printed - if the argument received
is a number (int or float) print using the .format style for float, if the argument is a string then print using the
.format style for float after converting to a float (use try/except block to handle error conditions gracefully)
3. define a function multiplePrint to receive multiple arguments to be printed (using args and kwargs), it will also
accept a keyword parameter called inputtype whose default value is "string". The inputtype parameter would be used
to decide to call the singleXXXPrint functions. If the value passed is "string" call singleStringPrint function
irrespective of the data type of the values to be printed. If is it is anything else call the singleNumericPrint.
(you may limit yourself to int, float and str type parameters)

Sample output given:

```
multiplePrint(10)
```

```
type received is: string
This is a actually a number: 10
```

```
multiplePrint('a')
```

```
type received is: string
This is a string: a
```

```
multiplePrint('a',10,12,x=10,y=20,z=30)
```

```
type received is: string
This is a string: a
This is a actually a number: 10
This is a actually a number: 12
This is a actually a number: 10
This is a actually a number: 20
This is a actually a number: 30
```

```
multiplePrint('a',10,12,inputtype='numeric',x=10,y=20,z=30)
```

```
type received is: numeric
a   was a string not printable as number
This is a number: 10.000000
This is a number: 12.000000
This is a number: 10.000000
This is a number: 20.000000
This is a number: 30.000000
```