1 BT-3051: Data Structure and Algorithms in Biology | Assignment 1

IMPORTANT:

You are allowed to discuss the problems verbally with your friends, but copying or looking at codes (either from your friend or the Web) is not permitted. Transgressions are easy to find, and will be reported to the "Sub-committee for the Discipline and Welfare of Students" and will be dealt with very strictly.

Instructions:

1. *Please download the notebook and modify the code blocks and the text blocks ONLY*
2. Reupload the .ipynb file to the google form shared.
3. **Deadline: 5 PM IST, 25.08.2023**
4. **Early Submission Bonus:** *Before 24 hrs.:* **10%**, *Before 48 hrs.:* **20%**
5. ** Late Submission Penalty:** *After 24 hrs.:* **20%**, *After 48 hrs.:* **40%**

Rules to write the code:

1. *Write Python code for the following questions using only basic Python data structures, loops, and conditional statements. You are not allowed to use any inbuilt function for searching, sorting, or replacing*
2. **Allowed:** *list, set, tuple, dictionary, variables(int, float), control statements(if-else, switch-case), loops(for, while)*
3. *Use the same function name and parameters given in each question*
4. *Write explanations as comments in the python file*

1.0.1 Question 1

Write a Python function that multiplies two numbers (a>b) but you are allowed to use only the addition and subtraction operator.

```
def mult(a,b):
  # Write your code here...
  return product

# Testcase
print(mult(5,3))
------------------------
>15
```

1. State the input size of your function.
2. Calculate the time complexity of your function and explain it.

(Explain it in a text block given below the question)

[ ]:
```
# Code block for Q1
```

Text block for Q1

1.0.2 Question 2

Many identification numbers have a certain structure to them. For example, the IIT Madras roll numbers are of the form:

“Alphabet-Alphabet-Num-Num-Alphabet-Num-Num-Num”.

Now write a python program that takes in a string as input and reports if the string is in concurrence with the structure.

Do this for: 1. The IITM ID (Structure above) 2. The PAN ID (Structure: AAAAANNNNM, where A is alphabet and N is a number)

```
def iitmid(id):
  # Write your code here...
  return True/False (as a bool, and not string)

def panid(pan):
  # Write your code here...
  return True/False (as a bool, and not string)

# Testcase
print(iitmid('BT20D700'))
print(panid('BT20D700'))
-------------------------------
>True
>False
```

Note:

Not allowed *regex*

[ ]:
```
# Code block for Q2
```

1.0.3 Question 3

Write a Python function that takes in a number (n<1000) and returns True if it is prime else False.

```
def isprime(num):
  # Write your code here...
  return True/False (as a bool, and not string)

#Testcase
```

```
print(isprime(7))
---------------------
>True
```

1. State the input size of your function.
2. Calculate the time complexity of your function and explain it.

(Explain it in a text block given below the question)

[ ]:
```
# Code block for Q3
```

Text block for Q3

1.0.4 Question 4

Write a Python function that takes two inputs L and R (0<L<=R<=1000) and returns the number of integers between L and R (inclusive) that satisfy this property:

The difference between the number and its reverse is a perfect square.

Ex: 3, 15, 40 and 56 are four such numbers satisfying the property.

```
def countsq(L, R):
  # Write your code here
  return intcount

# Testcase
print(countsq(1,12))
--------------------------
>12
```

[ ]:
```
# Code block for Q4
```

1.0.5 Question 5

ACP Pradyuman, a brilliant detective (pun-intended), is known for solving code puzzles. A programmer named Kamesh, seeks help with a decryption challenge inspired by the "CID" TV series. ACP Pradyuman creates a code question based on Kamesh's request.

Write a Python function `decrypt_message(message)` that reverses the characters within each word of a given string while preserving word order

Note:

Not allowed any inbuilt-function like *string.split* or *string.reverse*

```
def decrypt_message(message):
  # Write your code here...
  return decoded_message

# Testcase
message = 'Hello world'
decoded_message = decrypt_message(message)
```

```
print(decoded_message)
-------------------------
> 'olleH dlrow'
```

Application:

Reversing characters within words finds application in cryptography by adding complexity to encoding methods. This technique can be used to create unique transformations within encoded messages, enhancing the security of ciphers and making decryption more challenging without the proper key.

```
[ ]: # Code block for Q5
```