Assignment 01

Instructions

1. Write your functions according to the signature provided below in a python(.py) file and submit them to the autograder for evaluation.
2. The autograder has a limited number of attempts. Hence, test it thoroughly before submitting it for evaluation.
3. Save your python file as text(.txt) file and submit it to the canvas.
4. If the assignment has theoretical questions, upload those answers as a pdf file on the canvas.
5. Submission on the canvas is mandatory and the canvas submitted text file should be the same as that of the final submission of the autograder. If not, marks will be deducted.
6. No submission on canvas or no submission on autograder then marks will be deducted.
7. Access the auto grader at https://c200.luddy.indiana.edu.

Question 1

Prove or disprove the following questions on asymptotic notation. Provide detailed explanations to convey your ideas clearly.

You can either solve them over paper, scan and upload the pdf file over canvas.
Or
Type the solutions using latex and upload the compiled pdf file over canvas.

a. Prove or disprove that $f(n) \in \Theta(g(n))$ where $f(n) = 64n^2$ and $g(n) = n^4$

b. Prove or disprove that $f(n) \in \Omega(g(n))$ where $f(n) = 1/3n^2 + 10n - 2$ and $g(n) = n^2$

c. Prove or disprove that $f(n) \in O(g(n))$ where $f(n) = 3,00,000n^3 + 1$ and $g(n) = n^4$

d. Prove or disprove that $f(n) \in o(g(n))$ where $f(n) = 15n^{15}$ and $g(n) = n^{15}$

e. Prove or disprove that $f(n) \in \omega(g(n))$ where $f(n) = n^{20} - 17$ and $g(n) = n^{16}$

Question 2

Find the time complexities of the following functions. Provide detailed solutions.

a.
```
def function1(n):
  for i in range (0, n):
    for j in range(0, i+1):
      print("*")
      break
  return
```

b.
```
def function2(n):
  i = 1
  while i**2 <= n:
    i = i+1
```

```
  return
```

c.

```
def function3(m, n):
    while (m != n):
        if (m > n):
            m = m - n
        else:
            n = n - m
    return
```

d.

```
def function4(n):
  i = 1
  while i<n:
    j = n
    while(j>0):
      j = j//2
    i = 2*i
  return
```

e.

```
def function5(n):
  for i in range(0,n//2):
    for j in range(1,n-(n//2)+1):
      m=1
      while m<=n:
        m*=2
  return
```

Question 3

Baburao Ganpatrao Apte (or Babu Bhaiyya) lost all his money because Raju invested it in some scam, again. In dire need of cash Babu Bhaiyya joins Royal Circus as a part-time worker, he did ofcourse lie on his resume. On his first day of work a supervisor comes to him and says "Go bring me juggling balls from that box over there, also make sure that there are at least p balls of the same color. The count of other colored balls is of no particular concern." Babu Bhaiyya on looking at the label on the box realizes that there are c1, c2, c3, c4, and c5 number of red, yellow, blue, orange, and purple colored juggling balls respectively. But then after glancing into the box Babu Bhaiyya mutters "Ohh, I forgot I'm color blind". But does Babu Bhaiyya go to his supervisor and tell her that? ofcourse not he comes up with a solution some way or the other. Could you please help him? How many minimum number of balls should he give to his supervisor so her condition is met?

Examples

Example 1:

```
input: c1 = 3, c2 = 4, c3 = 3, c4 = 9, c5 = 23, p = 1
output: 1
```

Explanation: p=1 means at least one ball of same color, so just get one ball as it will be minimum.

Example 2:

```
input: c1 = 3, c2 = 4, c3 = 3, c4 = 9, c5 = 23, p = 2
output: 6
```

Explanation: p=2 means at least two balls of same color, so if we get 6 balls we can make sure atleast 2 of them will be of same color as there are only 5 colors.

Constraints

c1,c2,c3,c4,c5,p > 0

Function

```
def minimum_balls(c1, c2, c3, c4, c5, p):
  # Write your code
  return total_balls_taken_out
```

Question 4

Ken is renovating his straight entryway. Ken's current entryway has two color tiles, pink and blue. He wants to change all the plates to blue but is only allowed to change at max k number of plates. Return the number of blue tiles present in the longest running sequence of blue tiles that ken would see after installing.

Examples

Example 1:

```
Input:
(["blue","blue","blue","pink","pink","pink","blue","blue","blue","blue","pink"], k = 2)
Output: 6
```

Explanation: ["blue","blue","blue","pink","pink",**"blue"**,"blue","blue","blue","blue",**"blue"**] The Bolded blue tiles were changed from pink tiles to blue tiles by Ken to get the longest running sequence of blue tiles

Example 2:

```
Input: (["pink","pink","blue","blue","pink","blue","blue","pink"],k = 1)
Output: 5
```

Explanation: ["pink","pink","blue","blue",**"blue"**,"blue","blue","pink"] The Bolded blue tiles were changed from pink tiles to blue tiles by Ken to get the longest running sequence of blue tiles

Constraints

1. Expected time complexity is O(n)

Function

```
function def longestBlues(tiles: List[int], k) -> int:
  """
  Add your Code here
  """
  # return the number of  largest continous sequence of blue tiles
  return number_of_tiles
```

Question 5

In the whimsical land of Candyville, there is a diligent candy distributor named Tim. Tim distributes candies among the kids in the town every day. He offers 26 different flavors of candies, each recognized with a lowercase letter of the alphabet. Tim is very generous and allows kids to take whichever flavor and however many candies they desire. However, each kid is allowed to choose candies of only one flavor, with no restrictions on the number of candies they can take. Tim keeps a log of the candies being distributed. The log is in the form of a string, such as 'b4d9c18a24c9…'. This log can be interpreted as follows: Kid 1 took 4 candies of flavor 'b', Kid 2 took 9 candies of flavor 'd', and so on. At the end of the day, Tim wants to determine three things: 1) the total number of kids who received candy from him, 2) the total number of candies distributed on that day, and 3) the total number of candies distributed for each flavor. The desired output should be a string in a specific order: 'KxTyz', where 'x' is the numeric answer to 1), 'y' is the numeric answer to 2), and 'z' is a string, such as 'a13b4c9…', representing different flavors of candies followed by their respective counts in alphabetical order.

Examples

Example 1:

```
Input: 'a1a5b2c3'
Output: 'K4T11a6b2c3'
```

Explanation: There are 4 kids, 11 candies in total, and the substring 'a6b2c3' represents different flavored candies along with their respective counts, listed in alphabetical order.

Example 2:

```
Input: 'c18d4d13b6a14c5'
Output: 'K6T60a14b6c23d17'
```

Explanation: There are 6 kids, 60 candies in total, and the substring 'a14b6c23d17' represents different flavored candies along with their respective counts, listed in alphabetical order.

Constraints

- Input string contains only lower case alphabets and digits

Function

```
def CandiesLog(s):
  #Add your code here
  return res
```

Question 6

In the Kingdom of Asgard, there exists a mystical chain of knowledge known as the "Linked List". This chain is guarded by Heimdall, a gallant knight renowned for his problem-solving prowess. The Linked List has nodes, each of which contains a piece of the kingdom's ancient wisdom.

One fateful day, a prophecy was foretold: "To protect the kingdom from an impending doom, the chain of knowledge must be reordered. Every 'k' nodes of the Linked List must be reversed, lest the wisdom they contain unleash chaos upon the land!". However, if the number of nodes at the end was not a multiple of 'k', the wisdom in those nodes should remain untouched, for they contain the final secrets for maintaining the balance of Asgard.

The knight took it upon himself to reverse the segments of the Linked List according to the prophecy. Will you assist Heimdall in his quest? Can you write the Python spell, ahem, I mean code, to reverse every 'k' nodes in the Linked List?

Examples

Example 1:

```
hq = HeimdallQuest()
head = hq.create_linked_list([1, 2, 3, 4, 5])
new_head = hq.reverse_k_steps(head,2)
print(hq.get_linked_list(new_head))

Input:[1, 2, 3, 4, 5], k=2
Output:[2, 1, 4, 3, 5]
```

Explanation: Hiemdall starts with the chain of knowledge: 1 -> 2 -> 3 -> 4 -> 5. Upon chanting your spell with k=2, the chain should transform into: 2 -> 1 -> 4 -> 3 -> 5

Example 2:

```
Input:[76,23,22,65,34], k=6
Output:[76,23,22,65,34]
```

Explanation: Hiemdall starts with the chain of knowledge: 76 -> 23 -> 22 -> 65 -> 34. Upon chanting your spell with k=6, since the number of nodes is less than 6 the chain should transform the same

Constraints:

- Do not use array or any other data structures during transformation except linked list
- Return the final linked list as an array(list)

Function

```
class Node:
    def __init__(self, value):
        self.value = value
        self.next = None

class HeimdallQuest():
    def reverse_k_steps(self,head, k):
      #Write your code here
      return head

    def get_linked_list(self,head):
      #Write your code here to return list
      return res

    def create_linked_list(self,lst):
      #Write your code here to create linked list and return head of Linked list
      return head
```