CSA0888 – PYTHON PROGRAMMING

ASSIGNMENT – 5

NAME:V.MANOKARTHIK

REG.NO:192225077

1.LENGTH OF LAST WORD

INPUT:

```
def length_of_last_word(s):
    words = s.split()
    if not words:
        return 0
    last_word = words[-1]
    return len(last_word)
input_string = "my name is "
result = length_of_last_word(input_string)
print(result)
```

2.EMPLOYEE

INPUT:

```
def calculate_salary_with_bonus(salary, grade):
    bonus_percentage = 0
    if salary < 10000:
        bonus_percentage += 0.02
    if grade == 'A':
        bonus_percentage += 0.05
    elif grade == 'B':
        bonus_percentage += 0.1
    bonus = salary * bonus_percentage
    final_salary = salary + bonus
    return final_salary
```

```
salary = float(input("Enter the salary: "))
grade = input("Enter the grade (A or B): ")
final_salary = calculate_salary_with_bonus(salary, grade)
print(f"The final salary of the employee is ${final_salary:.2f}")
```

3.SQUARE

INPUT:

```
def numSquares(n):
    dp = [float('inf')] * (n + 1)
    dp[0] = 0
    squares = [i * i for i in range(1, int(n**0.5) + 1)]
    for i in range(1, n + 1):
        for square in squares:
            if i >= square:
                dp[i] = min(dp[i], dp[i - square] + 1)
    return dp[n]
n = 12
print(numSquares(n))
```

4.PRODUCT AND SUM

INPUT:

```
def can_divide(prod, summ):
    return prod % summ == 0
t = int(input("Enter the number of test cases: "))
for _ in range(t):
    prod, summ = map(int, input("Enter two integers (prod summ): ").split())
    if can_divide(prod, summ):
        print("YEAH")
    else:
        print("NAH")
```

5. RETURN THE INDEX TO ANY OF THE PEAKS

INPUT:

```
def find_peak_element(nums):
    left, right = 0, len(nums) - 1
    while left < right:
        mid = left + (right - left) // 2
        if nums[mid] > nums[mid + 1]:
            right = mid
        else:
            left = mid + 1
    return left
nums = [1, 2, 3, 1]
print(find_peak_element(nums))
```

6. IMPLEMENT A TRIANGULAR IN PASCAL

INPUT:

```
n = 5
for i in range(1, n+1):
        for j in range(0, n-i+1):
                print(' ', end='')
        C = 1
        for j in range(1, i+1):
                print(' ', C, sep='', end='')
                C = C * (i - j) // j
        print()
```

7.LONGEST SUBSTRING WITH AT LEAST K REPEATING CHARACTERS

INPUT:

```
def longest_substring(s, k):
    if len(s) == 0 or k > len(s):
        return 0
```

```
    char_count = {}
    for char in s:
        if char in char_count:
            char_count[char] += 1
        else:
            char_count[char] = 1
    for char, count in char_count.items():
        if count < k:
            return max(longest_substring(sub_s, k) for sub_s in s.split(char))
    return len(s)
s = "aaabb"
k = 3
print(longest_substring(s, k))
```

8.INPUT:

```
def min_swaps_to_chessboard(board):
    n = len(board)
    row_count = [0] * n
    col_count = [0] * n
    for i in range(n):
        for j in range(n):
            if board[i][j] != (i + j) % 2:
                return -1
            row_count[i] += board[i][j]
            col_count[j] += board[i][j]
    row_diff = abs(row_count[0] - n // 2)
    col_diff = abs(col_count[0] - n // 2)
    if row_diff > 1 or col_diff > 1:
        return -1
    row_swap = sum(board[0][i] != i % 2 for i in range(n))
```

```
    col_swap = sum(board[i][0] != i % 2 for i in range(n))
    if n % 2 == 0:
        return min(row_swap, n - row_swap) // 2 + min(col_swap, n - col_swap) // 2
    else:
        if row_count[0] > n // 2:
            row_swap = n - row_swap
        if col_count[0] > n // 2:
            col_swap = n - col_swap
        return row_swap // 2 + col_swap // 2
board = [
    [0, 1, 1],
    [1, 0, 0],
    [0, 1, 1]
]
print(min_swaps_to_chessboard(board))
```

9.INPUT:

```
def shuffle(l1, l2):
    shuffled = []
    min_len = min(len(l1), len(l2))
    for i in range(min_len):
        shuffled.append(l1[i])
        shuffled.append(l2[i])
    if len(l1) > len(l2):
        shuffled.extend(l1[min_len:])
    else:
        shuffled.extend(l2[min_len:])
    return shuffled
l1 = [1, 2, 3, 4]
l2 = ['a', 'b', 'c']
```

```
result = shuffle(l1, l2)
print(result)
```

10.INPUT:

```
def reverse_words(s):
    words = [word for word in s.split(' ') if word]
    reversed_words = words[::-1]
    reversed_string = ' '.join(reversed_words)
    return reversed_string
input_string = "  Hello  world!   "
output = reverse_words(input_string)
print(output)
```