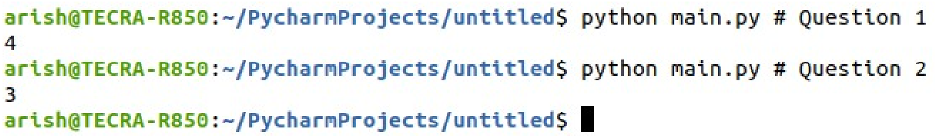
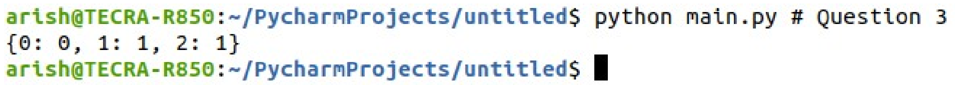
```
# Question 1

def countsubstring(str1, str2):
    flag = 0

    if len(str2) < len(str1):
        return 0

    if str1 in str2[0:len(str1)]:
        flag = 1

    return flag + countsubstring(str1, str2[1:])


print(countsubstring("ab", "abababab"))


# Question 2

def count(fun, arr):
    c = 0

    for i in arr:
        if fun(i):
            c = c + 1

       return c


print(count(lambda x: x > 2, [1, 2, 3, 4, 5]))
```



```
arish@TECRA-R850:~/PycharmProjects/untitled$ python main.py # Question 1
4
arish@TECRA-R850:~/PycharmProjects/untitled$ python main.py # Question 2
3
arish@TECRA-R850:~/PycharmProjects/untitled$ 
```



```
arish@TECRA-R850:~/PycharmProjects/untitled$ python main.py # Question 3
{0: 0, 1: 1, 2: 1}
arish@TECRA-R850:~/PycharmProjects/untitled$ 
```

```
def combinations(members, negations):
    combs = []
    result = []
    for i in members:
        for j in members:
            if i == j:
                continue
            if (i, j) not in negations and (j, i) not in negations:
                if (j, i) not in combs:
                    combs.append((i, j))
    if len(combs) == 0:
        return
    x1, y1 = combs[0]
    result.append((x1, y1))
    for i in range(1, len(combs)):
        x2, y2 = combs[i]
        if (x1 != x2) and (y1 != y2):
            result.append((x1, y1))
        x1, y1 = x2, y2
    return result


def prog(elem_count, table_count, rules):
    elems = list(range(0, elem_count))
    tables = list(range(0, table_count))
    dict = {}
    if elem_count >= 1 and table_count >= 1:
        dict[0] = 0
        elems.remove(0)
        if elem_count == 1:
            return dict
        else:
            if rules:
                results = combinations(elems, rules)
                if results is None:
                    return False
                table = 1
                for result in results:
                    x, y = result
                    elems.remove(x)
                    elems.remove(y)
                    if not dict.get(x):
                        dict[x] = table
                        table = table + 1
                    if not dict.get(y):
                        dict[y] = dict[x]
                for i in elems:
                    dict[i] = table
                    table = table + 1
                res = {}
                for i in sorted(dict):
                    res[i] = dict[i]
                return res
            else:
                for i in elems:
                    dict[i] = 0
                res = {}
                for i in sorted(dict):
                    res[i] = dict[i]
                return res
    else:
        return False

print(prog(3, 2, [(1, 0), (2, 0)]))
```