Lab-IX

Hasan Amin
(374866)

CS-250
Data Structures and Algorithms

School of Natural Sciences

Contents

Task 1: 3

Task 2: 4

Task 3: 5

Task 4: 6

Task 1:

Code

```
gcd = lambda a,b : a if (b==0) else gcd(abs(b),abs(a)%abs(b))

print(gcd(2,3))
print(gcd(16,100))
print(gcd(0,100))
```

Output

```
1
4
100
```

Task 2:

Code

```
def element_of(x,mylist):
    for i in mylist:
        if isinstance(i,(list,tuple)):
            returned=element_of(x,i)
            if returned==True:
                return True
        else:
            if i==x:
                return True
    return False

def filter_by_depth(depth, nested_list):
    if depth < 0:
        raise ValueError("Depth should be a non-negative integer")
    elif depth==0:
        return []

    mylist=[
        filter_by_depth(depth - 1, el) if isinstance(el, list) else el
        for el in nested_list
    ]
    return [x for x in mylist if x != []]

print("Element of() Testing")
print(element_of(5, [1,2,3,4,5,6,7]))
print(element_of(7, [1,2,[3,4,[5,6]],[7]]) )
print(element_of(77, [1,2,[3,4,[5,6]],[7]]))

print("Filter by Depth() Testing")
print(filter_by_depth(0, [1,2,3]))
print(filter_by_depth(1, [1,2,3]) )
print(filter_by_depth(5, [1,2,3]) )
print(filter_by_depth(2, [1,2,[3,4,[5,6]],[7]]))
```

Output

```
Element of() Testing
True
True
False
Filter by Depth() Testing
[]
[1, 2, 3]
[1, 2, 3]
[1, 2, [3, 4], [7]]
```

Task 3:

Code

```
def Multiply(x,y):
    while x!=1:
        if x%2==0:
            x=x//2
            y=2*y
        else:
            x=(x-1)//2
            y=3*y

    return y

def expo(x,y):
    if y==1 or x==0:
        return x
    if y%2 ==0:
        return expo((x**2),y//2)
    else:
        return x*expo((x**2),(y-1)//2)
print("Multiply Testing")
print(Multiply(2,38))
print(Multiply(3,56))
print(Multiply(3,17))

print("Exponent Testing")
print(expo(2,5))
print(expo(3,4))
print(expo(7,3)
```

Output

```
Multiply Testing
76
168
51
Exponent Testing
32
81
343
```

Task 4:

Code

```
pascal=lambda n,k: 1 if k==0 or n==k else pascal(n-1,k-1)+pascal(n-1,k)


print(pascal(4,2))
print(pascal(5,3))
print(pascal(10,7))
```

Output

```
6
10
120
```