Assignment_1

October 30, 2021

[1]:
```
#1

def table(n):
    """
    This function prints multiplication tables from 1 to 10 for given number.
    """
    for i in range(1,11):
        print("{0} x {1} = {2}".format(n,i,n*i))

n = int(input("Type a number:"))
print("Multiplication Tables for {} are:".format(n))
table(n)
```

```
Type a number:40
Multiplication Tables for 40 are:
40 x 1 = 40
40 x 2 = 80
40 x 3 = 120
40 x 4 = 160
40 x 5 = 200
40 x 6 = 240
40 x 7 = 280
40 x 8 = 320
40 x 9 = 360
40 x 10 = 400
```

[6]:
```
#2

def prime(n):
    """
    This function will check whether the given number is prime.
    """
    for i in range(2,n):
        if n%i == 0:
            return False
    return True

print("Prime twins less than 1000 are: ")
```

```
for x in range(2,1000):
    if prime(x)==True and prime(x+2)==True:
        print("({0},{1})".format(x,x+2))
```

```
Prime twins less than 1000 are:
(3,5)
(5,7)
(11,13)
(17,19)
(29,31)
(41,43)
(59,61)
(71,73)
(101,103)
(107,109)
(137,139)
(149,151)
(179,181)
(191,193)
(197,199)
(227,229)
(239,241)
(269,271)
(281,283)
(311,313)
(347,349)
(419,421)
(431,433)
(461,463)
(521,523)
(569,571)
(599,601)
(617,619)
(641,643)
(659,661)
(809,811)
(821,823)
(827,829)
(857,859)
(881,883)
```

[2]:
```
#3

def pFactors(n):
    """
    This function will return List of prime factors of a given number.
    """
```

```
    factors=[]
    i=2
    while (n/i != 1):
        if n%i == 0:
            factors.append(i)
            n=n//i
        else:
            i=i+1
    if n>1:
        factors.append(i)
    return factors

num = int(input("Type the number:"))
pFactors(num)
```

```
Type the number:56
```

[2]:
```
[2, 2, 2, 7]
```

[4]:
```
#4

def fact(n):
    """
    This function returns the factorial of a given number.
    """
    if n==0|n==1:
        return n
    else:
        return n*fact(n-1)

print("Enter the number of objects n and common difference r:\n")
n = int(input("n:"))
r = int(input("r:"))
print("Number of Permutations:",fact(n)/fact(n-r))
print("Number of Combinations:",fact(n)/(fact(n-r)*fact(r)))
```

```
Enter the number of objects n and common difference r:

n:8
r:4
Number of Permutations: 1680.0
Number of Combinations: 70.0
```

[6]:
```
#5

def decToBin(n):
    if n>=1:
        decToBin(n//2)
```

```
    print(n%2 , end='')

num = int(input("Enter a Decimal number:"))
print("Binary equivalent of {} is:".format(num), end=' ')
decToBin(num)
```

```
Enter a Decimal number:10
Binary equivalent of 10 is: 01010
```

[8]:
```
#6

def cubeSum(n):
    """
    This function will returns the sum of cube of individual digits of the␣
 ↪given number.
    """
    sum=0
    while n>0:
        d = n%10
        sum += d**3
        n//=10
    return sum

def isArmstrong(n):
    if n == cubeSum(n):
        return "YES"
    else:
        return "NO"

def printArmstrong(n):
    """
    This function will return Armstrong numbers between 0 and given number.
    """
    n=n+1
    arm=[]
    for i in range(n):
        if isArmstrong(i)=="YES":
            arm.append(i)
    return arm

num = int(input("Type a number:"))
print("The cube sum of {} is:".format(num), cubeSum(num))
print("is {} an Armstrong number? :".format(num), isArmstrong(num))
print("Armstrong numbers between 0 and {}:".format(num),printArmstrong(num))
```

```
Type a number:153
The cube sum of 153 is: 153
is 153 an Armstrong number? : YES
```

```
Armstrong numbers between 0 and 153: [0, 1, 153]
```

[9]:
```python
#7

def prodDigits(n):
    """
    This function will return Product of the Digits.
    """
    p=1
    while(n>0):
        d=n%10
        p*=d
        n//=10
    return p

num = int(input("Type a number:"))
print("Product of the Digits is:{}".format(prodDigits(num)))
```

```
Type a number:86
Product of the Digits is:48
```

[10]:
```python
#8

def MDR(n):
    """
    This function will calculate MDR and MPersistence of a given number.
    """
    s=str(n)
    p=0

    while len(s)>1:
        s=str(prodDigits(int(s)))  #prodDigits() from #7
        p+=1
    return int(s),p


num = int(input("Type a number:"))
mdr,per=MDR(num)
print("MDR of {} is {} and its multiplicative persistence is {}".
 ↪format(num,mdr,per))
```

```
Type a number:56
MDR of 56 is 0 and its multiplicative persistence is 2
```

[11]:
```python
#9

def sumPDivisors(n):
    """
```

```
    This function will return Sum of Proper Divisors for the given number.
    """
    sum=0
    for i in range(1,n):
        if(n%i==0):
            sum+=i
    return sum

num = int(input("Type a number:"))
print("Sum of Proper Divisors of {} is".format(num),sumPDivisors(num))
```

```
Type a number:96
Sum of Proper Divisors of 96 is 156
```

[12]:
```
#10

def perfectNum(l,u):
    """
    This function prints perfect numbers.
    """
    for i in range(l,u):
        if i==sumPDivisors(i):    # sumPDivisors from #9
            print(i,end=' ')

lo = int(input("Enter the lower range:"))
up = int(input("Enter the upper range:"))
print("The perfect numbers between {} and {} are: ".format(lo,up),end='')
perfectNum(lo,up)
```

```
Enter the lower range:100
Enter the upper range:500
The perfect numbers between 100 and 500 are: 496
```

[16]:
```
#11

def amicableNum(l,u):
    """
    This function prints Amicable numbers.
    """
    for i in range(l,u+1):
            for j in range(i,u+1):
                if i!=j:
                    if sumPDivisors(i)==j and sumPDivisors(j)==i:    #␣
 ↪sumPDivisors from #9
                        print(i,j)


lo = int(input("Enter the lower range:"))
```

```
up = int(input("Enter the upper range:"))
print("The amicable numbers between {} and {} are: ".format(lo,up),end='')
amicableNum(lo,up)
```

```
Enter the lower range:1000
Enter the upper range:2000
The amicable numbers between 1000 and 2000 are: 1184 1210
```

[17]:
```
#12

def oddNum(n):
    if n%2 != 0:
        return n

num = range(0,100)
odd_num = list(filter(oddNum,num))
print(odd_num)
```

```
[1, 3, 5, 7, 9, 11, 13, 15, 17, 19, 21, 23, 25, 27, 29, 31, 33, 35, 37, 39, 41,
43, 45, 47, 49, 51, 53, 55, 57, 59, 61, 63, 65, 67, 69, 71, 73, 75, 77, 79, 81,
83, 85, 87, 89, 91, 93, 95, 97, 99]
```

[20]:
```
#13

def cubes(n):
    return n**3

num = range(0,10)
cube_list = list(map(cubes,num))
print(list(num))
print(cube_list)
```

```
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
[0, 1, 8, 27, 64, 125, 216, 343, 512, 729]
```

[21]:
```
#14

def evenNum(n):
    if n%2 == 0:
        return n

def cubes(n):
    return n**3

num = range(0,20)
even_num = list(filter(evenNum,num))
cube_list = list(map(cubes,even_num))
print(even_num)
print(cube_list)
```

```
[2, 4, 6, 8, 10, 12, 14, 16, 18]
[8, 64, 216, 512, 1000, 1728, 2744, 4096, 5832]
```