In [6]:
```
n=int(input())
f_count=0
for i in range(1,n+1):
    if(n%i==0):
        f_count+=1
if(f_count==2):
    print("Prime Number")
else:
    print("Not a Prime Number")

```

```
9
Not a Prime Number
```

In [12]:
```
 n1=int(input())
f_sum=0
for j in range(1,n1+1):
    if(n1%j==0):
        f_sum+=j
print("Factors sum =",f_sum)
if(f_sum==n1):
    print("Perfect Number")
else:
      print("Not a Perfect Number")
```

```
5
Factors sum = 6
Not a Perfect Number
```

In [7]:
```
s=int(input())
e=int(input())
e_s=e_c=0
print("Even Numbers are: ")
for k in range(s,e+1):
    if(k%2==0):
        print(k,end=' ')
        e_s=e_s+k
        e_c=e_c+1
print("\nEven Numbers Sum= ",e_s)
print("Even Numbers Count= ",e_c)
```

```
1
100
Even Numbers are:
2 4 6 8 10 12 14 16 18 20 22 24 26 28 30 32 34 36 38 40 42 44 46 48 50 52 54 56
58 60 62 64 66 68 70 72 74 76 78 80 82 84 86 88 90 92 94 96 98 100
Even Numbers Sum=  2550
Even Numbers Count=  50
```

In [12]:
```
#10 to 1 numbers
i=10
while(i>=1):
    print(i,end=' ')
    i=i-1
```

```
10 9 8 7 6 5 4 3 2 1
```

In [13]:
```
#1 to 10 numbers
i=1
while(i<=10):
    print(i,end=' ')
    i=i+1
```

```
1 2 3 4 5 6 7 8 9 10
```

In [21]:
```
num1=int(input())
rev=0
while(num1>0):
    r=num1%10
    rev=rev*10+r
    num1=num1//10
print("Reverse =",rev)
```

```
1001
Reverse = 1001
```

In [22]:
```
num1=int(input())
temp=num1
rev=0
while(num1>0):
    r=num1%10
    rev=rev*10+r
    num1=num1//10
if(temp==rev):
    print("Palindrom")
else:
    print("Not a Palindrom")
```

```
545
Palindrom
```

Function

It is a block of code which runs only when it is called

-Functions are divided into two types

1.Builtin functions
2.User defined functions

Again user defined functions are four types

1.With arguments with return value
2.with arguments without return value
3.without arguments with return value
4.without arguments without return value

syntax for functions

```
def functionname(arguments/parameters):
        statements
function_name(arguments/parameters)
```

Advantages of using functions

1.Reuse of code
2.Improving the clarity of code
3.Dividing the large program code into smaller pieces

In [25]:
```
#with arguments with return value
a,b=int(input()),int(input())
def add1(a,b):
    return a+b
add1(a,b)
```

```
6
6
```

Out[25]: 12

In [29]:
```
#with arguments without return value
a=int(input())
b=int(input())
def add2(a,b):
    print(a+b)
add2(a,b)
```

```
6
6
12
```

In [30]:
```
#without arguments with return value
v,m=2,5
def add3():
    return v+m
add3()
```

Out[30]: 7

In [31]:
```
#without arguments without return value
v,m=2,5
def add3():
    print(v+m)
add3()

```

```
7
```

In [35]:
```
def even_odd(n):
    if(n%2==0):
        print("Even")
    else:
        print("Odd")
n=int(input())
even_odd(n)
```

```
5
Odd
```

In [ ]:
```
def prime(num):
    f_c=0
    for i in range(1,num+1):
        if(num%i==0):
            f_c+=1
        if(f_c==2):
            print("prime")
        else:
            print("Not a prime")
num=int(input())
prime(num)
```

In [ ]:
```
s1,e1=int(input()),int(input())
def prime_range(s1,e1):
    for j in range(s1,e1+1):
        if(prime(j)==True):
            print(j,end=' ')
prime_range(s1,e1)
```

In [19]:
```
def perfect(s):
    f_sum=0
    for i in range(1,s):
        if(s%i==0):
            f_sum+=i
        if(f_sum==s):
            print("perfect number")
    else:
        print("Not a perfect number")
s=int(input())
perfect(s)
```

```
28
perfect number
perfect number
perfect number
perfect number
perfect number
perfect number
perfect number
perfect number
perfect number
perfect number
perfect number
perfect number
perfect number
perfect number
perfect number
Not a perfect number
```

In [ ]:
```
def perfect_range(s1,e1):
    for j in range(s1,e1+1):
        if(perfect(j)==True):
            print(j,end=' ')
s1,e1=int(input()),int(input())
perfect_range(s1,e1)

```

```
# Strings
    -string is an collection of characters
    -
```

In [ ]:
```
a="welcome"
```

In [2]:
```
a1=input()

```

```
Welcome
```

In [6]:
```
print(len(a1))
print(type(a1))
print(min(a1))
print(max(a1))
print(sorted(a1))
```

```
7
<class 'str'>
W
o
['W', 'c', 'e', 'e', 'l', 'm', 'o']
```

In [7]:
```
a1

```

Out[7]: 'Welcome'

In [76]:
```
print(a1[0:3])

```

```
Wel
```

In [12]:
```
print(a1[2:5])
```

```
lco
```

In [13]:
```
print(a1[0::3])
```

```
Wce
```

In [14]:
```
print(a1[::-1])
```

```
emocleW
```

In [15]:
```
d='welcome'
d.capitalize()
```

Out[15]: 'Welcome'

In [17]:
```
d.count('o')
```

Out[17]: 1

In [18]:
```
d="Welcome to python programmming"
d.count("o")
```

Out[18]: 4

In [19]:
```
k1="PYTHON"
print(k1.lower())
```

```
python
```

In [20]:
```
k2="python"
print(k2.upper())
```

```
PYTHON
```

In [24]:
```
d1="PYTHON PROGRAMMING"
print(d1.index('O'))
print(d1.rindex('O'))
```

```
4
9
```

In [26]:
```
#find()
print(d1.find("O"))
```

```
4
```

In [27]:
```
print(d1.rfind("O"))
```

```
9
```

In [28]:
```
k="PYTHON"
L="python"
print(k.isupper())
print(L.islower())
```

```
True
True
```

In [29]:
```
h="English"
h.swapcase()
```

Out[29]: 'eNGLISH'

In [37]:
```
# isalpha(),isalnum(),isdigit(),isspace()
b="jupyter"
print(b.isalpha())
b1="jupyter"
print(b.isalnum())
b12="1288474"
print(b12.isdigit())
n=" "
print(n.isspace())
```

```
True
True
True
True
```

In [38]:
```
# starts with ends with
f,l="hi hello gd evng","apssdc"
print(f.startswith("hi"))
print(f.startswith('g'))
print(l.endswith("c"))
```

```
True
False
True
```

In [39]:
```
# replace
v1="day"
v1.replace("d","s")
```

Out[39]:
```
'say'
```

In [41]:
```
# title()
m='good afternoon'
print(m.title())
print(m.istitle())
```

```
Good Afternoon
False
```

In [42]:
```
#split()
v="this is srk college located at vijayawada"
print(v.split())
```

```
['this', 'is', 'srk', 'college', 'located', 'at', 'vijayawada']
```

In [43]:
```
g="py@th@on"
print(g.split())
print(g.split("@"))
```

```
['py@th@on']
['py', 'th', 'on']
```

In [45]:
```
# join()
print("#".join(g))
print("@".join(g))
```

```
p#y#@#t#h#@#o#n
p@y@@@t@h@@@o@n
```

In [52]:
```
#strip(),lstrip(),rstrip()
h1,h2,h3="    lenovo ","    hii    ","hello    "
h1.strip
```

Out[52]: 'lenovo'

In [53]:
```
h2.lstrip()
```

Out[53]: 'hii   '

In [54]:
```
h3.rstrip()
```

Out[54]: 'hello'

In [58]:
```
#centre
fg="siddhardha"
fg.center(30)
```

Out[58]: '   siddhardha   '

In [59]:
```
#zfill()
fg.zfill(20)
```

Out[59]: '0000000000siddhardha'

In [63]:
```
student="Siddhardha"
for ch in student:
    print(ch,end=" ")
```

```
S i d d h a r d h a
```

In [68]:
```
word="ApsSdC@123"
for i in word:
    if(i.isupper()):
        print(i,end=' ')
```

```
A S C
```

In [75]:

```
w=input()
up=lw=dig=sp=""
for k in w:
    if(k.isdigit()):
        dig=dig+k
    elif(k.isupper()):
            up=up+k
    elif(k.islower()):
                lw=lw+k
    else:
        sp+=k
        print("Uppercase letters are: ",*up)
        print("\nLowercase Letters are: ",lw)
        print("\nDigits are: ",dig)
        print("\nSpecial characters are: ",sp)

```

```
SDFGH12345dfef@#$$%$
Uppercase letters are:  S D F G H

Lowercase Letters are:  dfef

Digits are:  12345

Special characters are:  @
Uppercase letters are:  S D F G H

Lowercase Letters are:  dfef

Digits are:  12345

Special characters are:  @#
Uppercase letters are:  S D F G H

Lowercase Letters are:  dfef

Digits are:  12345

Special characters are:  @#$
Uppercase letters are:  S D F G H

Lowercase Letters are:  dfef

Digits are:  12345

Special characters are:  @#$$
Uppercase letters are:  S D F G H

Lowercase Letters are:  dfef

Digits are:  12345

Special characters are:  @#$$%
Uppercase letters are:  S D F G H

Lowercase Letters are:  dfef
```

```
Digits are:  12345

Special characters are:  @#$$%$
```

In [ ]:
```
1
```

In [ ]:
```
1
```

In [ ]:
```
1
```

In [ ]:
```
1
```

In [ ]:
```
1
```