In [9]:

```
string_input = str(input("Enter a string of characters : "))
words = string_input.split()
dictionary = {}
for word in words:
    if (word[0] not in dictionary.keys()):
        dictionary[word[0]] = []
        dictionary[word[0]].append(word)
    else:
        if (word not in dictionary[word[0]]):
            dictionary[word[0]].append(word)

print(dictionary)
```

```
Enter a string of characters : rohith
{'r': ['rohith']}
```

In [10]:

```
n=int(input("Enter a number: "))
a=[]
while(n>0):
    dig=n%2
    a.append(dig)
    n=n//2
a.reverse()
print("Binary Equivalent is: ")
for i in a:
    print(i,end=" ")
```

```
Enter a number: 125
Binary Equivalent is:
1 1 1 1 1 0 1
```

In [*]:

```
fname = input("Enter file name: ")
with open(fname, 'r') as f:
    for line in f:
        l=line.title()
print(l)
```

```
Enter file name:
```

In [14]:

```
str=input("Enter a String : ")
dict = {}
for i in str:
    dict[i] = str.count(i)
print (dict)
```

```
Enter a String : rohtih
{'r': 1, 'o': 1, 'h': 2, 't': 1, 'i': 1}
```

In [18]:

```
def add(datatype, *args):
    if datatype =='int':
        answer = 0
    if datatype =='str':
        answer =''

    for x in args:
        answer = answer + x
    print(answer)

add('int', 5, 15,2000)

add('str', 'Hi ', 'First batch of ML students')
```

```
2020
Hi First batch of ML students
```

In [19]:

```
class A:
    def fun1(self):
        print('feature_1 of class A')
    def fun2(self):
        print('feature_2 of class A')

class B(A):

    def fun1(self):
        print('Modified feature_1 of class A by class B')
    def fun3(self):
        print('feature_3 of class B')


obj = B()

obj.fun1()
```

```
Modified feature_1 of class A by class B
```

In [3]:

```
import threading
def print_cube(num):
    """
    function to print cube of given num
    """
    print("Cube: {}".format(num * num * num))
def print_square(num):
    """
    function to print square of given num
    """
    print("Square: {}".format(num * num))
if __name__ == "__main__":
    t1 = threading.Thread(target=print_square, args=(10,))

    t2 = threading.Thread(target=print_cube, args=(10,))

    t1.start()

    t2.start()

    t1.join()

    t2.join()

    print("Done!")
```

```
Square: 100
Cube: 1000
Done!
```