Question1:

Given a string with words separated by spaces, create an acronym by taking the first letter of each word. The output string should be in upper case.

In [1]:
```
# Example:

Input="State Bank of India"
lst=Input.split(" ")
lst1=[]
for i in lst:
    lst1.append(i[0].upper())
str1="".join(lst1)
print(str1)
```

```
SBOI
```

Question 2:

Write a program that prints the sand clock pattern.

In [2]:
```
# Example:

nr=5 #it represents the length of 1st half of sand like pattern

for r in range(0,nr):
    print("*",end=" ")
print()
for r1 in range(1,nr-1):
    for s in range(r1):
        print(" ",end="")
    for c in range(1):
        print("*", end="")
    for s1 in range(1,2*nr-2*r1-2):
        print(" ",end="")
    for c1 in range(1):
        print("*", end="")
    print()
for r2 in range(1,nr-1):
    for s2 in range(r2,nr-1):
        print(" ",end="")
    for c2 in range(1):
        print("*", end="")
    for s3 in range(0,2*r2-1):
        print(" ",end="")
    for c3 in range(1):
        print("*", end="")
    print()
for r3 in range(0,nr):
    print("*",end=" ")
```

```
* * * * *
 *     *
  *   *
   * *
   * *
  *   *
 *     *
* * * * *
```

Question 3:

Given a list of integers, group all numbers which end with the same digit and return as a dictionary. The keys of the dictionary will be the last digit and value of dictionary will be a list of numbers. If given list is empty return -1.

In [3]:
```
# Example:

Input = [22,3,22,2,71,11]

def dict_return(Input):
    dict1={}
    lst_keys=[]
    for i in Input:
        x=i%10
        if x not in lst_keys:
            y=lst_keys.append(x)
    for j in lst_keys:
        lst=[]
        for values in range(len(Input)):
            if Input[values]%10==j:
                lst.append(Input[values])
                dict1[j]=lst
        lst=[]
    return dict1
print(dict_return(Input))
```

```
{2: [22, 22, 2], 3: [3], 1: [71, 11]}
```

Question 4:

Given a list of integers, find the index of the maximum value. If there are two maximums, return index of the first one. If given list is empty return -1. Do not use the index() method.

In [4]:
```
# Example 1:
Input = []


lst=[]
def returnmax():
    if len(Input)==0:
        return -1
    else:
        x=np.max(Input)
        for i in range(len(Input)):
            if Input[i] ==x:
                lst.append(i)
        return lst[0]


print(returnmax())
```

```
-1
```

In [6]:
```
# Example 2:

import numpy as np
Input = [56,67,43,32,1]


lst=[]
def returnmax():
    if len(Input)==0:
        return -1
    else:
        x=np.max(Input)
        for i in range(len(Input)):
            if Input[i] ==x:
                lst.append(i)
        return lst[0]


print(returnmax())
```

```
1
```

Question 5

Binary search is effective, since we all know that its time complexity is O(log n). But in large scale systems like Netflix, where there are trillions of megabytes to process, even this may fall short. In large scale systems like that of Netflix, data lookups are always parallel, simultaneously searching relevant node servers in a server cluster. The data at Netflix is stored redundantly in such a way that they can be accessed in the fastest way possible. So to compensate for this, one of the approaches they apply is to approximate search solutions inside their massive data store. A way to start that is to use Approximate Binary Search. This basically means that after a certain number of calls to itself, the binary search either returns the result or a value that says Not found here.As a developer, your task is to devise the algorithm. Assume that the data you are getting is sorted, but you dont know which direction.Thus, given an upper limit on the number of self-calls you can make, write a program that searches the data and returns either the index of where the data is, or -1 (indicating that the data is nowhere to be found within the given constraints.)

In [7]:
```
import numpy as np
Input=np.array([2, 4, 2, 1, 5, 6,8,9])
num=1
limit=2

def func(Input,num,limit):
        if np.median(Input)==num:
            return Input.index(num)
        else:
            for i in range(1,limit):
                h=np.hsplit(Input,2)
                for i in range(len(h)):
                    if np.median(h[i])==num:
                        return Input.index(num)
                    else:
                        return -1
print(func(Input,num,limit))
```

```
-1
```

Question 6

Given a list of numbers find the spread of the list. Here, spread = Max - Min

In [1]:
```
def spread(Input):
    maximum=0
    minimum=5
    for i in range(len(Input)):
        if Input[i]>maximum:
            maximum=Input[i]
    for j in range(len(Input)):
        if Input[j]<minimum:
            minimum=Input[j]
    return (maximum-minimum)

# Example1:
Input = [-1,0,5]

print(spread(Input))
```

```
6
```

In [ ]:
```

```