SNS COLLEGE OF ENGINEERING
Kurumbapalayam (Po), Coimbatore – 641 107
An Autonomous Institution
Accredited by NBA – AICTE and Accredited by NAAC – UGC with 'A' Grade
Approved by AICTE, New Delhi & Affiliated to Anna University, Chennai

DEPARTMENT OF COMPUTER SCIENCE AND DESIGN

19IT103 – COMPUTATIONAL THINKING AND PYTHON PROGRAMMING

- **A readable, dynamic, pleasant, flexible, fast and powerful language**

Recap:

- Simple strategies for developing algorithms:
  - Iteration
  - Recursion

- Iteration: A sequence that is executed repeatedly so long as a certain condition holds. A sequence of statements is executed until a specified condition is true is called iterations.
  - for loop
  - While loop

Recap:

- Simple strategies for developing algorithms:
  - Iteration
  - Recursion

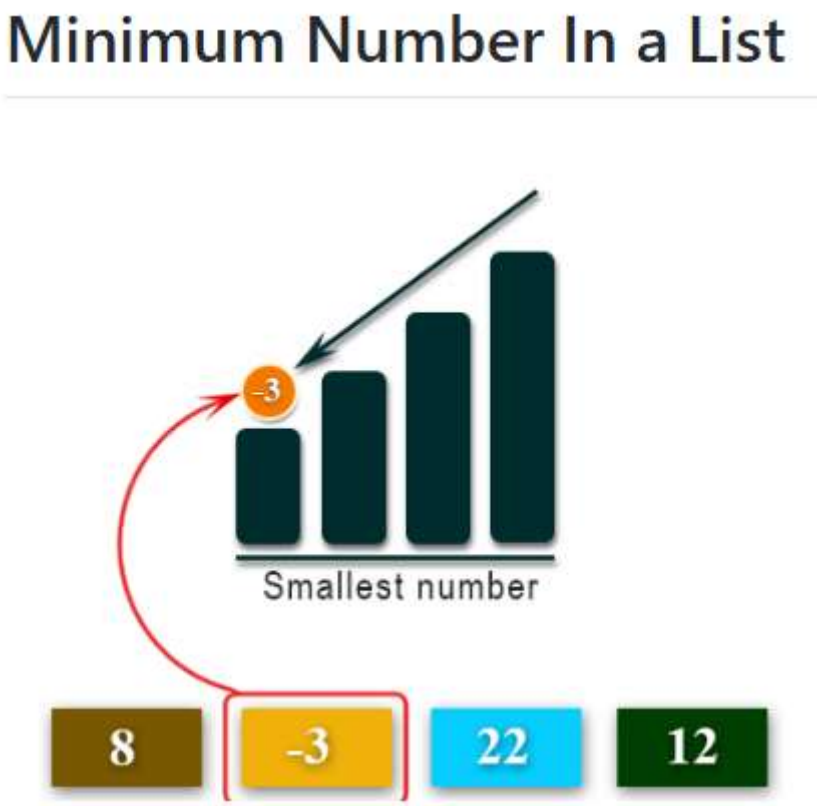
- Recursion: A function that calls itself is known as recursion.
- Recursion is a process by which a function calls itself repeatedly until some specified condition has been satisfied.

1.8 Illustrative problems:

- Find a minimum in a list
- insert a card in a list of sorted cards

1.8.1 Find a minimum in a list :

- To find the minimum element in the given list of elements.



1.8.1 Find a minimum in a list :

Problem Statement:

- The problem is to find the minimum element in the given list of elements. Finding minimum in a list of elements can be achieved in different ways.

1.8.1 Find a minimum in a list :

Different ways to find minimum element in a list:

- One way is to sort the list of elements in ascending order and get the first element as minimum.
- **Another method** is to compare each element with other.
  - As an initial step, first element of the list is considered as minimum element.
  - And in each iteration, each element in the list is compared with the minimum.
  - If the element in the list is less than the minimum then swap both elements else compare with the next element in the list.
  - These steps are continued until the end of the list and finally print the minimum.

1.8.1 Find a minimum in a list :

Find minimum of two numbers:

```
#find minimum of two numbers
# a and b are parameters''


def find_min(a, b):
    if a < b:
        return a
    return b


print("Enter two values :")
a = int(input())
b = int(input())
print("Minimum number is ", find_min(a, b))
```

1.8.1 Find a minimum in a list :

Find minimum of two numbers:

main.py

Run

Shell

```
1  def find_min(a,b):
2      if(a<b):
3          return a
4      return b
5
6  print("Enter two values:")
7  a=int(input())
8  b=int(input())
9  print("Minimum number is ",find_min(a,b))
10
```

```
Enter two values:
5
88
Minimum number is  5
>
```

1.8.1 Find a minimum in a list :

Find minimum of three numbers:

```
#find minimum of three numbers

def find_min(a, b):
    if a < b:
        return a
    return b

# a, b and c are parameters
def min_of_three(a, b, c):
    minVal = find_min(a, b)
    if c < minVal:
        return c
    return minVal


print("Enter three numbers: ")
a = int(input())
b = int(input())
c = int(input())

print("Minimum number is ", min_of_three(a, b, c))
```

1.8.1 Find a minimum in a list :

Find minimum of three numbers:

main.py

Run

```
def find_min(a,b):
    if(a<b):
        return a
    return b

def min_of_three(a,b,c):
    minVal=find_min(a,b)
    if c<minVal:
        return c
    return minVal

print("Enter three values:")
a=int(input())
b=int(input())
c=int(input())
print("Minimum number is ",min_of_three(a,b,c))

```

Shell

```
Enter three values:
77
3
56
Minimum number is  3
>
```

1.8.1 Find a minimum in a list :

Find minimum number in a list:

```
# find minimum of a list
def min_of_list(aList):
    if not aList:
        return None
    minVal = aList[0]
    for number in aList[1:]:
        if number < minVal:
            minVal = number
    return minVal


myList = []
limit = int(input("Enter the limit: "))
print("Enter the elements:\n")
for i in range(limit):
    element = int(input())
    myList.append(element)

print("Minimum of list is ", min_of_list(myList))
```

1.8.1 Find a minimum in a list :

Find minimum number in a list:

main.py

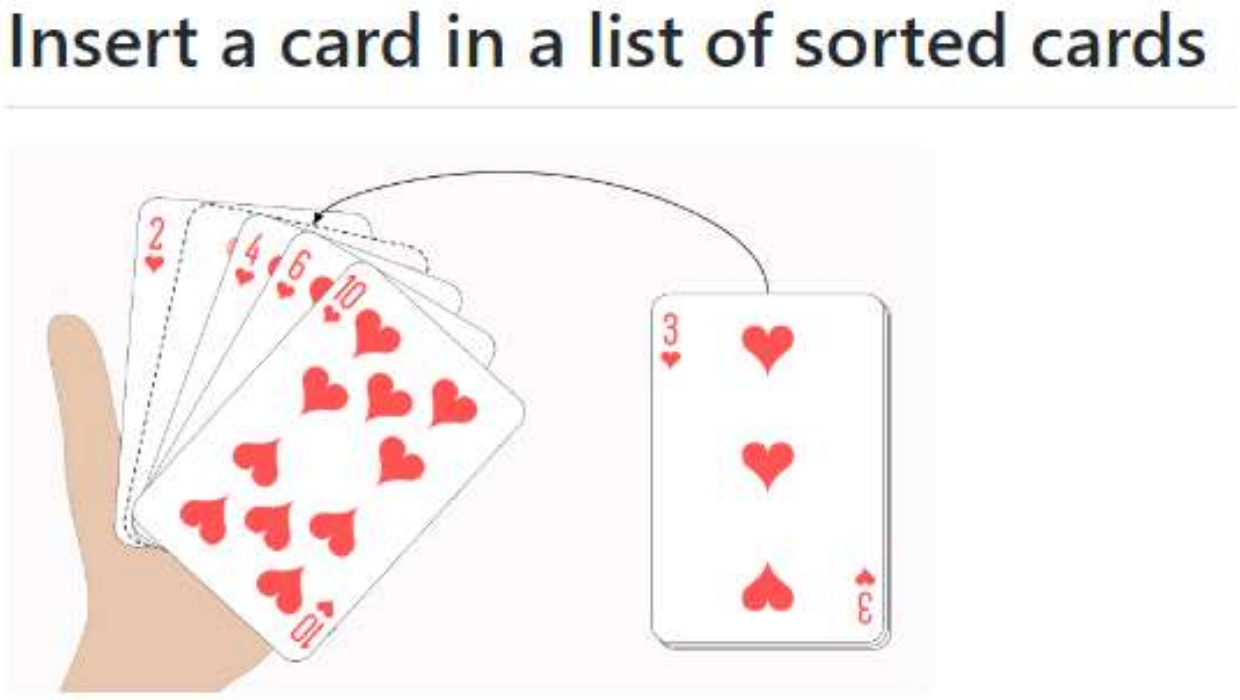
Run

```
def min_of_list(aList):
    if not aList:
        return None
    minVal = aList[0]
    for number in aList[1:]:
        if number < minVal:
            minVal = number
    return minVal

myList = []
limit = int(input("Enter the limit: "))
print("Enter the elements:\n")
for i in range(limit):
    element = int(input())
    myList.append(element)

print("Minimum of list is ", min_of_list(myList))
```

Shell

```
Enter the limit: 5
Enter the elements:

-1
5
-2
6
8
Minimum of list is  -2
>
```

1.8.2 insert a card in a list of sorted cards :



1.8.2 insert a card in a list of sorted cards :

Playing cards are one of the techniques of sorting and the steps are shown as follows:

- Start with an empty left hand and cards face down on the table.
- Then remove one card at a time from the table and Insert it into the correct position in the left hand.
- To find a correct position for a card, we compare it with each of the cards already in the hand from left to right.
- Once the position is found, the cards from that position are moved to the next higher indexed position and in that order.
- New card is inserted at the current position.

1.8.2 insert a card in a list of sorted cards :

```
order = {
    'A': 1, '2': 2, '3': 3, '4': 4,
    '5': 5, '6': 6, '7': 7, '8': 8,
    '9': 9, '10': 10,
    'J': 11, 'Q': 12, 'K': 13
}


def insertCard(deck, newCard):
    for card in deck:
        if order[card] > order[newCard]:
            index = deck.index(card)
            deck.insert(index, newCard)
            break
    return deck


deck = ['2', '5', '8', '10', 'J', 'K']  # initial set of cards
print("deck = ", deck)
newCard = input("Enter the new card to be inserted:")  # get the new card
insertCard(deck, newCard)
print(deck)
```

1.8.2 insert a card in a list of sorted cards :

Output 1:

```
order = {
    'A': 1, '2': 2, '3': 3, '4': 4,
    '5': 5, '6': 6, '7': 7, '8': 8,
    '9': 9, '10': 10,
    'J': 11, 'Q': 12, 'K': 13
}


def insertCard(deck, newCard):
    for card in deck:
        if order[card] > order[newCard]:
            index = deck.index(card)
            deck.insert(index, newCard)
            break
    return deck


deck = ['2', '5', '8', '10', 'J', 'K']  # initial set of cards
print("deck = ", deck)
newCard = input("Enter the new card to be inserted :")  # get the new card
```

```
deck =  ['2', '5', '8', '10', 'J', 'K']
Enter the new card to be inserted:Q
['2', '5', '8', '10', 'J', 'Q', 'K']
>
```

1.8.2 insert a card in a list of sorted cards :

Output 2:

main.py

Run

```
order = {
    'A': 1, '2': 2, '3': 3, '4': 4,
    '5': 5, '6': 6, '7': 7, '8': 8,
    '9': 9, '10': 10,
    'J': 11, 'Q': 12, 'K': 13
}


def insertCard(deck, newCard):
    for card in deck:
        if order[card] > order[newCard]:
            index = deck.index(card)
            deck.insert(index, newCard)
            break
    return deck

```

Shell

```
deck =  ['2', '5', '8', '10', 'J', 'K']
Enter the new card to be inserted:7
['2', '5', '7', '8', '10', 'J', 'K']
>
```

Summary:

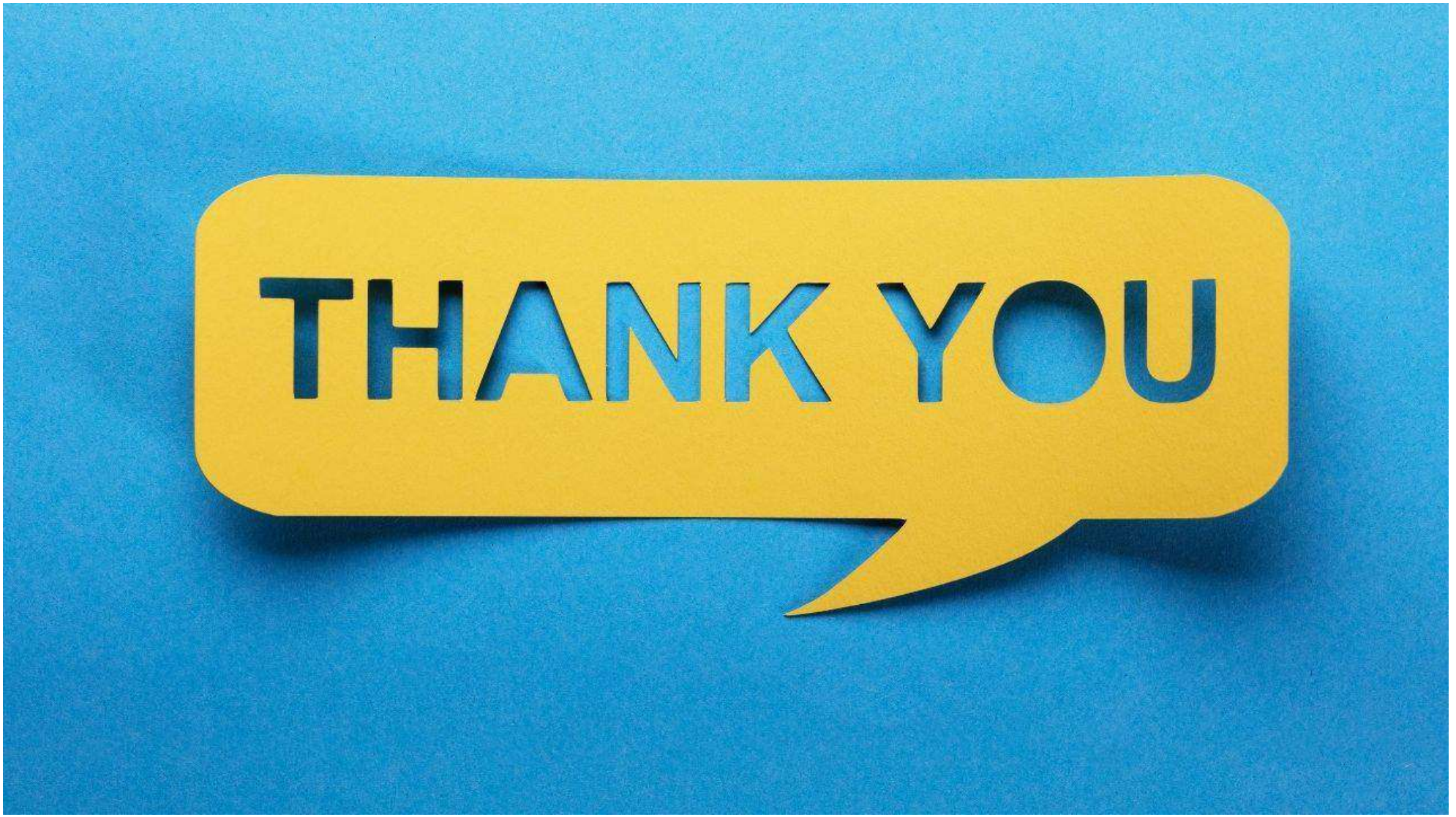
1. Find a minimum in a list :

- One way is to sort the list of elements in ascending order and get the first element as minimum.
- Another method is to compare each element with other.
  - As an initial step, first element of the list is considered as minimum element.
  - And in each iteration, each element in the list is compared with the minimum.
  - If the element in the list is less than the minimum then swap both elements else compare with the next element in the list.
  - These steps are continued until the end of the list and finally print the minimum.

Summary:

2. insert a card in a list of sorted cards :

- Start with an empty left hand and cards face down on the table.
- Then remove one card at a time from the table and Insert it into the correct position in the left hand.
- To find a correct position for a card, we compare it with each of the cards already in the hand from left to right.
- Once the position is found, the cards from that position are moved to the next higher indexed position and in that order.
- New card is inserted at the current position.


THANK YOU