unit 22 stack

In [ ]:
```
#quiz 1
# output

['Banana', 'Apple', 'Strawberry']
```

In [ ]:
```
#quiz 2

# output

[10, 30, 50, 70, 90]
```

In [4]:
```
#qn 1
import re

def display_html(html):
    stack = []
    output = []
    tokens = re.findall(r'<[^>]+>|[^<]+', html)
    for token in tokens:
        if token.startswith('<'):
            if not token.startswith('</'):
                stack.append(token)
            else:
                stack.pop()
        else:
            text = token.strip()
            if text:
                if stack and stack[-1] == "<li>":
                    output.append("• " + text)
                else:
                    output.append(text)
    for line in output:
        print(line)
html_doc = """
<h1>Hello, World!</h1>
<p>We are learning the art of coding
with Python programming language.</p>
<ul>
<li>Data Structures</li>
<li>Algorithms</li>
<li>and Computational Thinking</li>
</ul>
"""

display_html(html_doc)
```

```
Hello, World!
We are learning the art of coding
with Python programming language.
• Data Structures
• Algorithms
• and Computational Thinking
```

In [ ]:
```python

```

In [ ]:
```python
# unit 23 queue
```

In [ ]:
```python
#quiz 1
# output

['Tomato', 'Strawberry', 'Grapes']
```

In [ ]:
```python
#quiz 2
# output  = []
```

In [5]:
```python
# qn 1

class Deque:
    def __init__(self):
        self.queue = []

    def add_first(self, item):
        self.queue.insert(0, item)

    def remove_first(self):
        if len(self.queue) > 0:
            return self.queue.pop(0)

    def add_last(self, item):
        self.queue.append(item)

    def remove_last(self):
        if len(self.queue) > 0:
            return self.queue.pop()

d = Deque()
d.add_first(10)
d.add_last(20)
d.add_first(5)

print("Current queue:", d.queue)
print("Removed from front:", d.remove_first())
print("Removed from rear:", d.remove_last())
print("Final queue:", d.queue)
```

```
Current queue: [5, 10, 20]
Removed from front: 5
Removed from rear: 20
Final queue: [10]
```

In [ ]:

```

```

In [ ]:

```
# unit 24
# sequential search

this algorithm scans the list at one go to find both the maximum and minimum v
each element with the current largest and smallest elements.
For the given list,the largest value found is 59 and the smallest value is 11.
Therefore, when the program prints the result, the output is 59 11.
```

In [ ]:

```
# quiz 2

The loop runs 7 times,and in each iteration up to two comparisons are made:
one to check for a new maximum and one to check for a new minimum.
Hence, in the worst case, the algorithm performs 14 comparisons in total.
```

In [ ]:

```

```

In [6]:

```python
# qn1

def word_count(S, x):
    c = 0
    for i in S:
        if i == x:
            c += 1
    return c

S = input("Input a sentence: ").split()
x = input("Input a word to search: ")
count = word_count(S, x)
print(f"In S, {x} is appeared in {count} times.")
```

```
In S, name is appeared in 3 times.
```

In [ ]:

```

```

unit 25

binary search

In [ ]:

```
# quiz 1

6 counts will be printed.
```

In [ ]:

```
# quiz 2
2 counts will be printed.
```

In [8]:

```python
# qn 1
```

```
def search_insert_position(nums, x):
    for i in range(len(nums)):
        if nums[i] >= x:
            return i
    return len(nums)

nums = [10, 20, 40, 50, 60, 80]
x = int(input("Input a number to insert: "))
pos = search_insert_position(nums, x)
print(f"{x} should be inserted at position {pos}.")
nums.insert(pos, x)
print(nums)
```

```
55 should be inserted at position 4.
[10, 20, 40, 50, 55, 60, 80]
```

In [ ]:

unit 26

hash table

quiz 1

key: 1655 hashkey : 1655 mod 8 = 7

In [ ]:

quiz 2

The Little Prince: Slot 2 The Old Man and the Sea: Slot 7 The Little Mermaid: Slot 3 eauty and the Beast: Slot 1 The Last Leaf: Slot 7 Alice in WonderLand: Slot 5

In [9]:

```
# qn 1

def int_to_roman(num):
    table = {1000:'M', 900:'CM', 500:'D', 400:'CD',
             100:'C', 90:'XC', 50:'L', 40:'XL',
             10:'X', 9:'IX', 5:'V', 4:'IV', 1:'I'}
    result = ""
    for value in table:
        while num >= value:
            result = result + table[value]
            num = num - value
    return result

num = int(input("Input a number: "))
print(int_to_roman(num))
```

```
MCMXCIX
```

```
In [ ]:
```