Python Programming

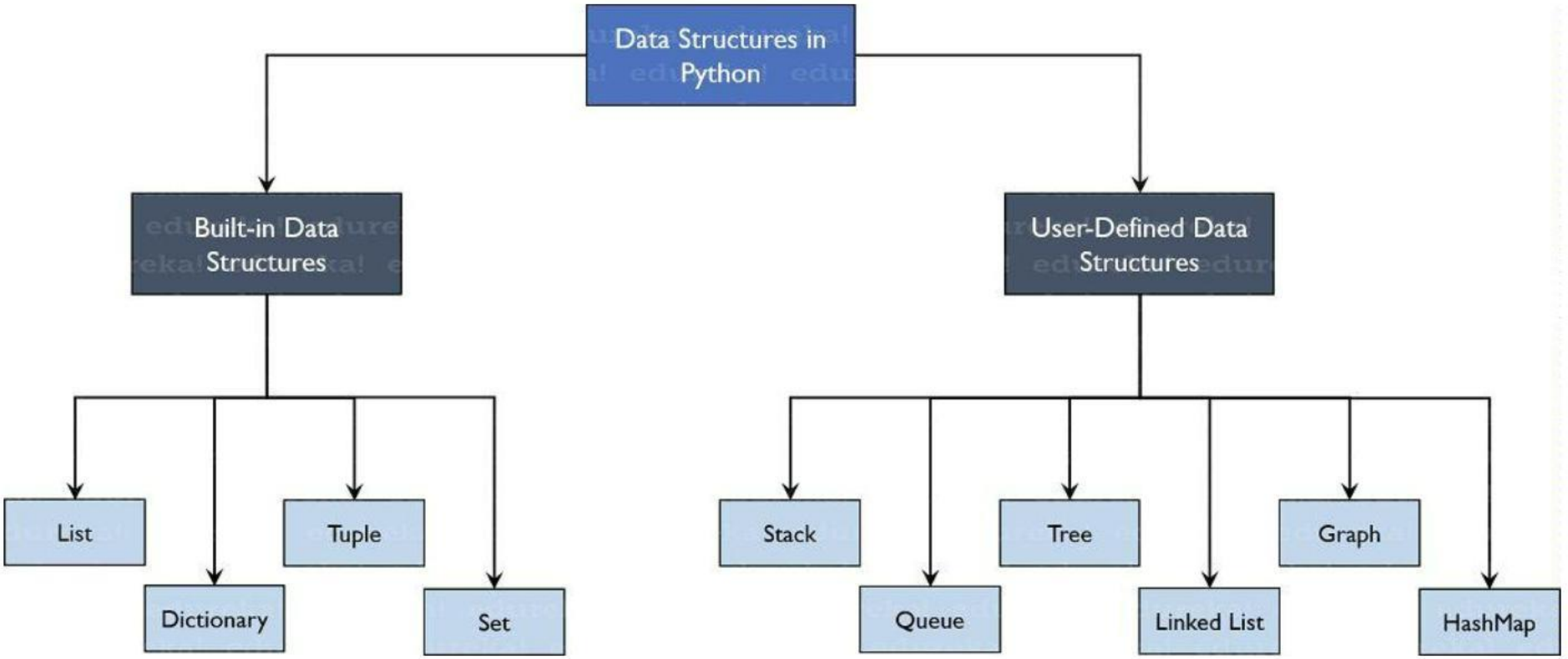
Лекц 10

Багш Ж.Золжаргал
Х . Хулан

What is Data Structure?

- Өгөгдлийг зохион байгуулах, удирдах, хадгалахад илүү хялбар, үр дүнтэй, өөрчлөлт хийх боломжийг олгодог тул чухал үүрэг гүйцэтгэдэг.
- Өгөгдлийн бүтэц нь өгөгдлийн цуглуулгыг хадгалах, хооронд нь уялдуулах, тэдгээрийн дагуу үйлдлийг гүйцэтгэх боломжийг бүрдүүлж өгөгдлөө цэгцлэх боломжийг олгодог.
- Паятон дээрх өгөгдлийн бүтэцүүдийн төрлүүд
  - List, Tuple, Dictionary, Set

Data Structures in Python

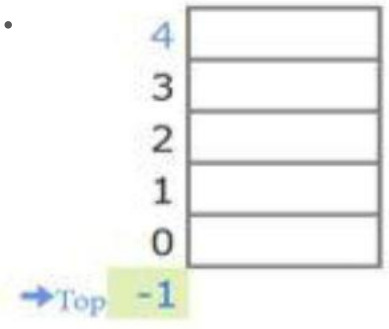


Array vs Lists

- Жагсаалт нь нэг төрлийн бус өгөгдлийн элементийг хадгалах боломжийг олгодог бол массивууд нь зөвхөн нэг төрлийн элементүүдийг хадгалах боломжийг олгодог.

Stack

- Стек нь шугаман өгөгдлийн бүтэц бөгөөд сүүлд орсон нь эхэлж гарах зарчимтай. (LIFO)
- Сүүлд нэмэгдсэн элемент рүү эхэлж хандах боломжтой
- Үүнийг массивийн бүтцээр гүйцэтгэдэг
  - Push (нэмэх) элемент
  - Pop (устгах) элемент
- Элементэд зөвхөн дээрээс нь хандана.



Stack

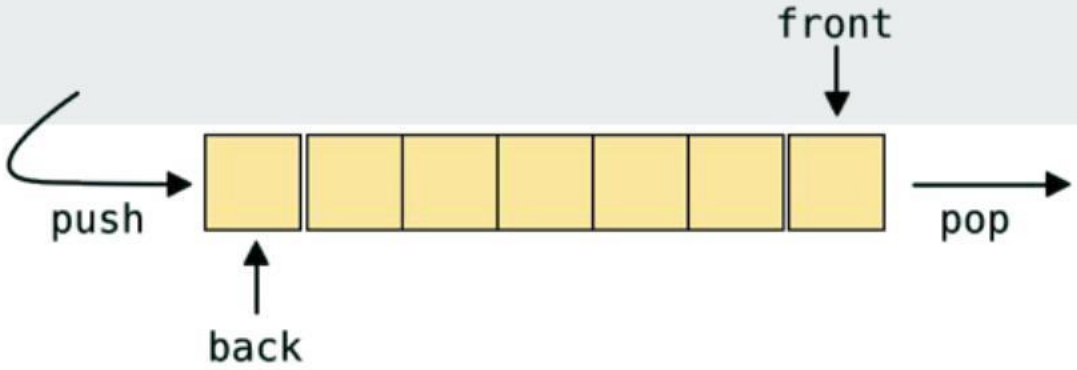
```
# Python code to demonstrate Implementing
# stack using list
stack = ["Amar", "Akbar", "Anthony"]
stack.append("Ram")
stack.append("Iqbal")
print(stack)

# Removes the last item
print(stack.pop())

print(stack)

# Removes the last item
print(stack.pop())

print(stack)
```

Stack

- Үр дүн

```
['Amar', 'Akbar', 'Anthony', 'Ram', 'Iqbal']
Iqbal
['Amar', 'Akbar', 'Anthony', 'Ram']
Ram
['Amar', 'Akbar', 'Anthony']
```

Queue



- Дараалал нь бас шугаман өгөгдлийн бүтэц бөгөөд FIFO зарчим дээр суурилсан.
  - Эхэлж орсон элемент рүү эхэлж хандана
  - Дарааллын эхлэл болон төгсгөлөөс үйлдэл хийнэ
    - Толгой-сүүл, урд тал - ард тал
  - Элемент нэмэх болон утсгах үйлдэлтэй
    - En-queue, De-queue
- Хэрэглээ
  - Сүлжээний буффер
  - Замын хөдөлгөөний түгжрэл

Queue

```
# Python code to demonstrate Implementing
# Queue using list
queue = ["Amar", "Akbar", "Anthony"]
queue.append("Ram")
queue.append("Iqbal")
print(queue)

# Removes the first item
print(queue.pop(0))

print(queue)

# Removes the first item
print(queue.pop(0))

print(queue)
```

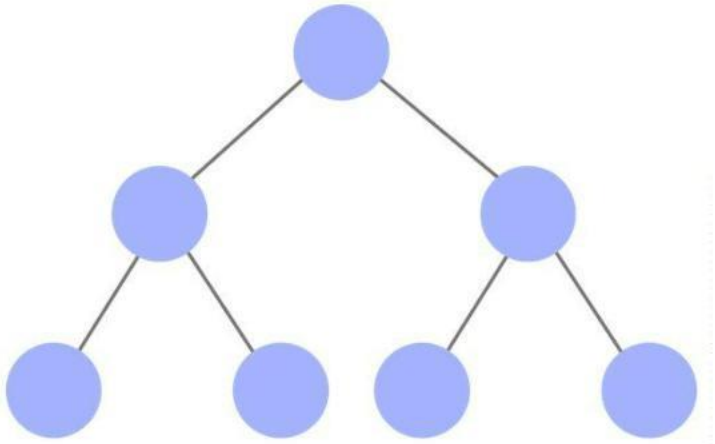
Queue

- Үр дүн

```
['Amar', 'Akbar', 'Anthony', 'Ram', 'Iqbal']
Amar
['Akbar', 'Anthony', 'Ram', 'Iqbal']
Akbar
['Anthony', 'Ram', 'Iqbal']
```

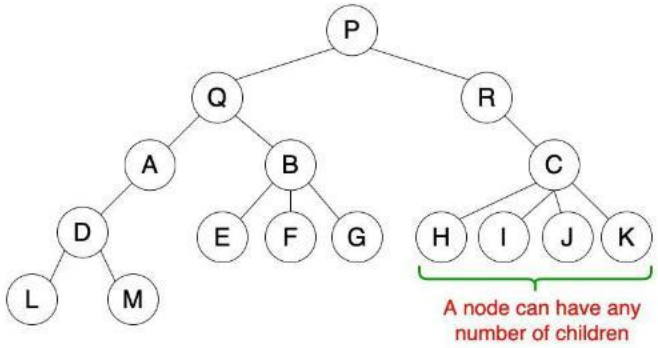
Tree

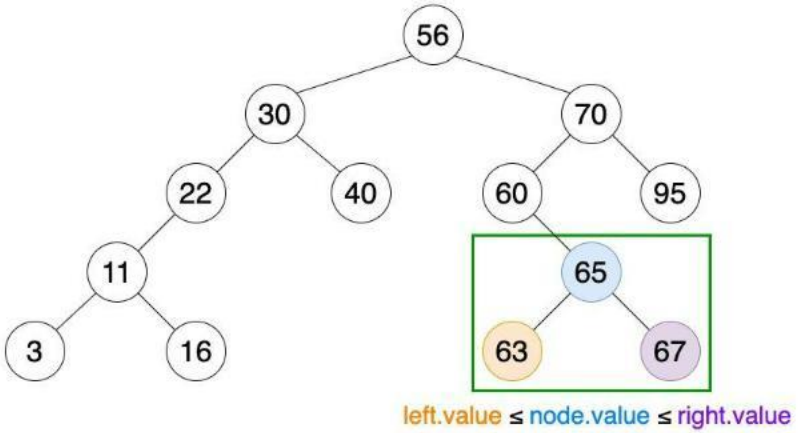
- Мод нь шугаман биш өгөгдлийн бүтэц бөгөөд үндсэн зангилаатай
- Үндэс нь бас зангилаа бөгөөд
- Өмнөх зангилааг эцэг, дараах зангилааг хүү гэнэ.
  - Сүүлийн зангилаануудыг навч гэнэ.
- Мод нь шаталсан бүтцийг хэрэгжүүлдэг
  - HTML хуудас
  - Энэ нь бас хайлтыг илүү дүнтэй болгодог

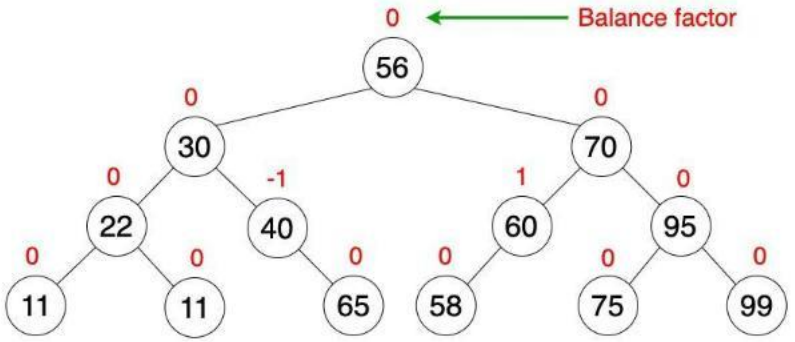


Tree

- General tree
- Binary tree
- Binary search tree
- AVL tree
- Red-black tree
- Splay tree
- Treap
- B-tree







Binary Tree

- Зангилааг тодорхойлох

```
# The Node Class defines the structure of a Node
class Node:
    # Initialize the attributes of Node
    def __init__(self, data):
        self.left = None # Left Child
        self.right = None # Right Child
        self.data = data # Node Data
```

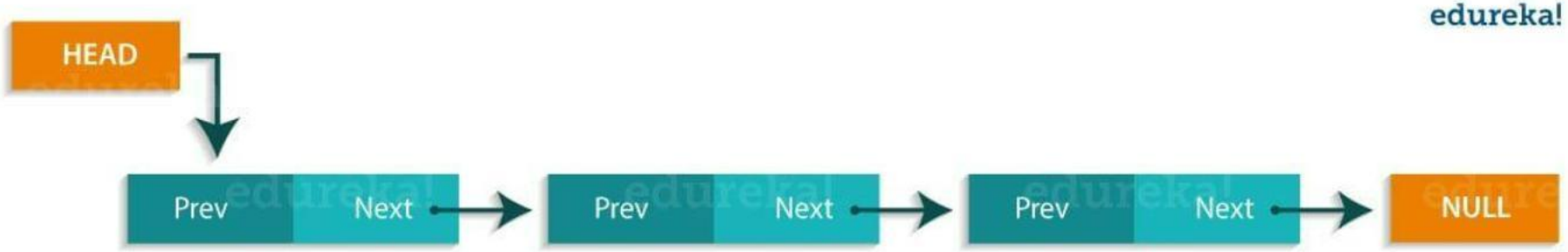
```python
class Node:
    def __init__(self, data):
        self.left = None
        self.right = None
        self.data = data

root = Node(10) # Instantiating the Tree
# Tree Structure
#        10
#      /    \
#    None   None

root.left = Node(34) # Setting the left child of the root to 34
root.right = Node(89) # Setting the right child of the root to 89

# Tree Structure
#          10
#        /    \
#       34      89
#     /    \  /    \
#  None  None None None
```

```python
def preorder(node):
    if node:
        # Print the value of the root node first
        print(node.data)

        # Recursively call preorder on the left subtree until we
        preorder(node.left)

        # Recursively call preorder on the right subtree until we
        preorder(node.right)

# For the tree,
#          10
#        /    \
#       34      89
#     /    \  /    \
#  20      45  56    54

# Preorder traversal: 10 34 20 45 89 56 54
```

```
def postorder(node):
    if node:
        # Recursively call postorder on the left subtree until we
        postorder(node.left)

        # Recursively call postorder on the right subtree until we
        postorder(node.right)

        # Print the value of the root node
        print(node.data)

# For the tree,
#          10
#        /    \
#       34      89
#     /    \  /    \
#  20      45  56    54

# Postorder traversal: 20 45 34 56 54 89 10
```

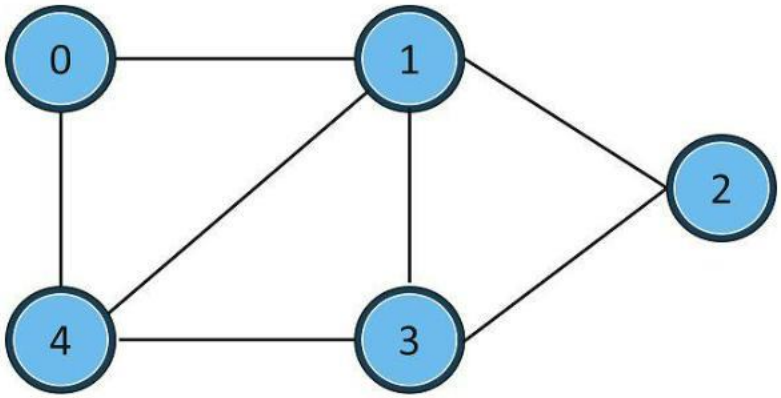
Linked list

- Холбоос жагсаалт нь шугаман өгөгдлийн бүтэц ба тэдгээр нь дараалсан биш бөгөөд заагч ашиглан өөр хоорондоо холбогддог.
- Нэг зангилаа нь өгөгдөл болон дараагын элементийн заагчаас бүтнэ.
- Хэрэглээ
  - Зураг үзүүлдэг апп
  - Хөгжим тоглуулагч



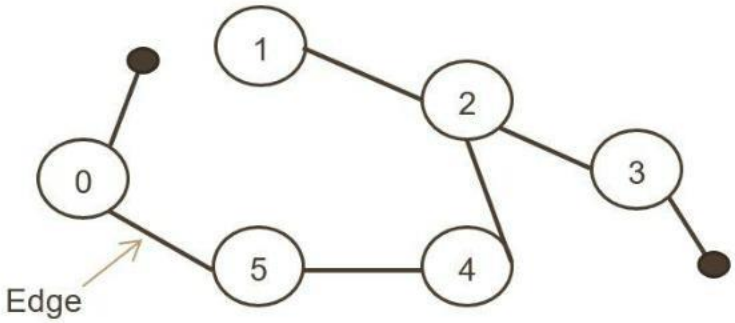
Graph

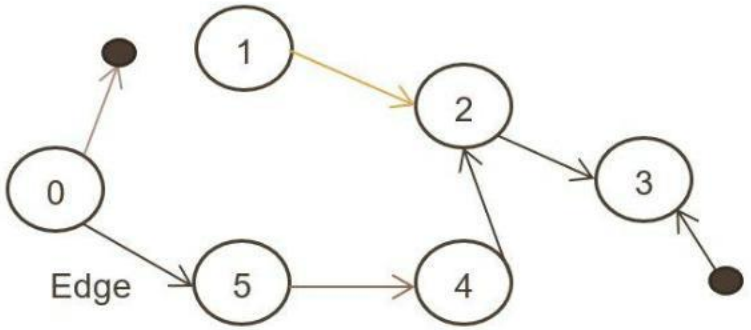
- Граф нь орой (node) ба ирмэг (edge) гэж нэрлэгддэг цэгүүдийн мэдээллийг цуглуулахад ашиглагддаг.
- Граф нь дэлхийн газрын зургыг хамгийн зөв дүрсэлж чадна.
- Зангилаа гэж нэрлэгдэх өгөгдлийн цэгүүдийн хоорондох зай
  - Зайг олох
  - Хамгийн бага замыг олох
- Хэрэглээ
  - Google map
  - Uber



Graph

- Жинтэй граф (Weighted)
- Жингүй граф (Unweighted)
- Чиглэлгүй граф (Undirected)
- Чиглэлтэй граф (Directed)





HashMaps

- HashMaps нь паятон дээрх dictionary-тай ижилхэн.
- Хэрэглээ
  - Phonebook