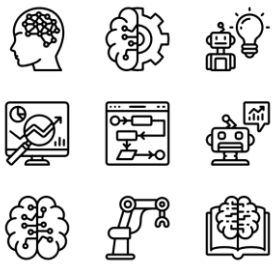
Computer Science for Practicing Engineers

Stack và Queue



TS. Huỳnh Bá Diệu
Email: dieuhb@gmail.com
Phone: 0914146868

1

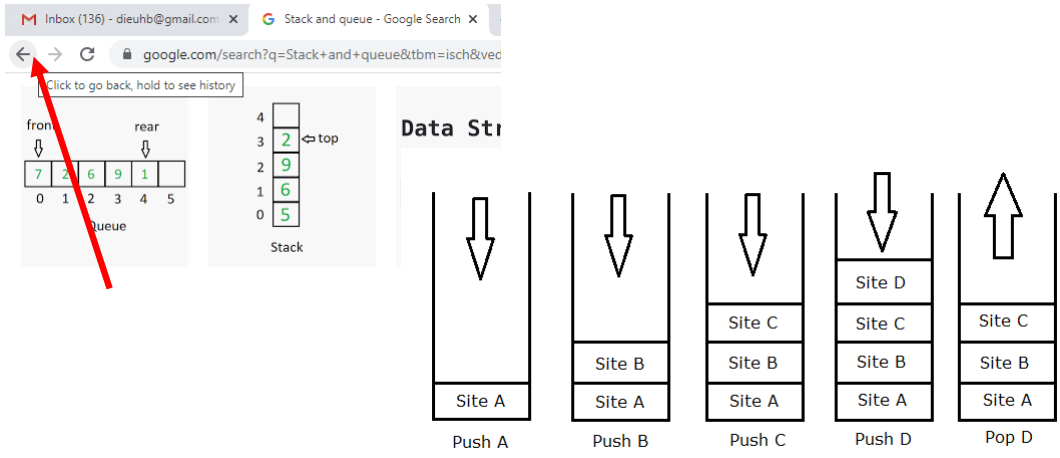
Nội dung

1. Ngăn xếp
2. Hàng đợi và hàng đợi ưu tiên
3. Các ứng dụng của ngăn xếp và hàng đợi

2

2

Ngăn xếp



3

Ngăn xếp (Stack)

- Các phần tử cùng kiểu
- Số phần tử không cố định
- Các phần tử được đưa vào và lấy ra ở đỉnh ngăn xếp
- Ngăn xếp hoạt động theo cơ chế LIFO (Last In First Out)

Ví dụ:

- Chồng đĩa khi rửa + Chồng đĩa khi đưa ra phục vụ

4

Ngăn xếp (Stack)

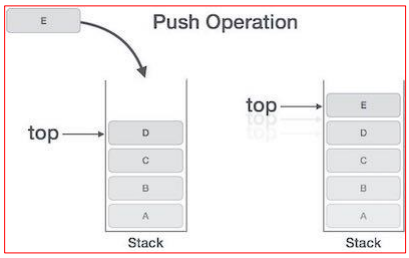
Có 4 thao tác cơ bản trên Stack

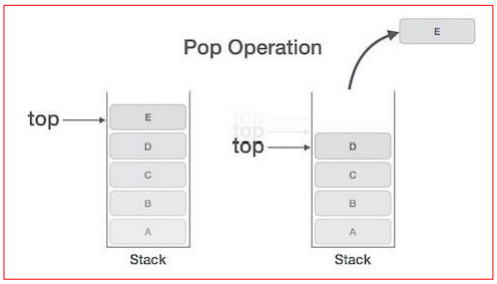
Tạo mới ngăn xếp;

Kiểm tra ngăn xếp có rỗng không;

Thêm 1 phần tử vào đỉnh ngăn xếp;

Lấy 1 phần tử ở đỉnh ngăn xếp ra.





5

Ngăn xếp (Stack)

Có thể tự định nghĩa kiểu Stack hoặc dùng kiểu có sẵn trong ngôn ngữ lập trình.

import java.util.Stack; Stack <String> S; **S= new Stack<>()**;

SN	Methods with Description
1	**boolean empty()**
2	***Object peek( )***
3	**Object pop( )**
4	**Object push(Object element)**
5	***int search(Object element)***

6

```
import java.util.Stack;
public class CSPE_MyP12_Using_StackType {
    Stack <String> S;
    void them()    {
        S= new Stack <>();
        for(int i =1; i<=10; i++)     S.push(" Phan tu thu " + i);
        System.out.println("\n Noi dung Stack: \n");
        while (!S.empty())   {  String x= S.pop();   System.out.println(" --" + x); }
        System.out.println("\n Noi dung Stack: "+ S);
    }
      public static void main(String[] args) {
        CSPE_MyP12_Using_StackType m= new CSPE_MyP12_Using_StackType();
        m.them();
    }
}
```

7

Ứng dụng của ngăn xếp

Dùng kiểu dữ liệu Stack, viết chương trình kiểm tra chuỗi T chứa các ký tự "(", ")" có tạo thành chuỗi hợp lệ không.

()
()()
((()))
()((()))
)(
((((
())(

Giải pháp của bạn???

8

Ứng dụng của ngăn xếp

Dùng kiểu dữ liệu Stack, viết chương trình kiểm tra chuỗi T chứa các ký tự "(", ")" có tạo thành chuỗi hợp lệ không.

Thuật toán:

1. Tạo ngăn xếp S rỗng
2. Duyệt từng phần tử x của chuỗi T
   - Nếu x là dấu (thì cho vào ngăn xếp S
   - Ngược lại Nếu x là dấu) thì kiểm tra ngăn xếp có rỗng không. Nếu rỗng thì trả về false ngược lại lấy ra 1 phần tử từ S

3. Nếu ngăn xếp rỗng thì trả về true ngược lại trả về false

9

Find-duplicate-parenthesis-expression

Cho chuỗi S biểu diễn một biểu thức. Kiểm tra xem chuỗi S có chứa cặp ngoặc thừa hay không?

Ví dụ:

(a+b)*(c-a): không có

(a+b) + z: không có

((a+b))*(c-a): có

Giải pháp của bạn???

10

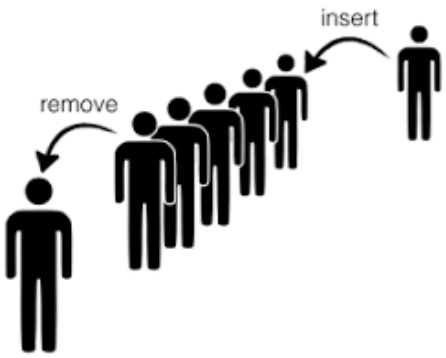
Ứng dụng ngăn xếp

Khử đệ qui bài toán tháp Hà Nội

- **Cách lưu trữ 1 phần tử trong ngăn xếp???**
- **Giải thuật để khử đệ qui như thế nào???**

11

Hàng đợi (Queue)



12

Hàng đợi (Queue)

- Các phần tử cùng kiểu
- Số phần tử không cố định
- Các phần tử được đưa vào ở đuôi hàng đợi và lấy ra ở đầu hàng đợi
- Hàng đợi hoạt động theo cơ chế FIFO (First In First Out)

Ví dụ:

- Xếp hàng vô thang máy
- Máy in trong mạng

13

Các thao tác Cơ bản trên Hàng đợi (Queue)

Có bốn thao tác chính:

- Tạo hàng đợi rỗng
- Kiểm tra hàng đợi có rỗng hay không
- Thêm 1 phần tử có giá trị x vào đuôi hàng đợi
- Lấy 1 phần tử ở đầu hàng đợi ra

[kiểm tra hàng đợi đã đầy chưa]

Có thể dùng mảng hoặc danh sách liên kết để cài đặt hàng đợi.

14

Các thao tác Cơ bản trên Hàng đợi (Queue)

Một phần tử trong hàng đợi:

```
class Node
{
        int data;
        Node next;
        Node(int x) {data=x; next = null;}
        Node(int x, Node t) {data=x; next = t;}
}
```

15

Cài đặt Hàng đợi (Queue)

```
public class MyQueue
{
        Node head, tail;
        MyQueue() { head= tail= null;}
        boolean EmptyQ()
                {return head==null;}
        void AddQ(int x)
        {
                Node t= new Node (x);
                if(head== null) head= tail=t;
                else { tail.next = t; tail = t;}
        }
```

```
        int RemoveQ()
        {
                int x= 0;
                if(head==null)
                System.out.println("Hang doi rong");
                else
                {
                   x= head.data;
                    if(head==tail) head=tail=null;
                   else   head= head.next;
                  return x;
                }
        }
```

16

Cài đặt Hàng đợi (Queue)

```
public static void main( String argvs)
{
        MyQueue Q= new MyQueue();
        for(int i=1; i<=10; i++) Q.AddQ(i);
        System.out.println("\n cac so trong hang doi:");
        while (!Q.EmptyQ()) System.out.println(Q.RemoveQ() + " ");
}
```

17

```
int getk(int k)
   {
      int x=0;
      MyQueue t= new MyQueue();
      int vt=1;
      // lay k=1 phan tu o Q bo sang t
      while(!EmptyQ() && vt<k) { t.AddQ(RemoveQ()); vt++; }
       if(vt==k) x= RemoveQ(); // lay phan tu thu k neu nhu hang doi con
       else System.out.println("\n Hang doi khong du " + k+ " phan tu!");
      while(!EmptyQ() ) t.AddQ(RemoveQ()); // lay cac phan tu con lai trong Q bo vao t
      while(!t.EmptyQ() ) AddQ(t.RemoveQ());  // Lay cac phan tu tu t bo lai hang doi Q
     return x;
   }
```

18

Sử dụng Hàng đợi của Java

```
import java.util.*;
public class Dung_Queue {
public static void main(String[] args) throws InterruptedException {
        int time = 30;
        Queue<Integer> Q = new LinkedList<Integer>();
        for (int i = time; i >= 0; i--) queue.add(i);
        while (!Q.isEmpty())
            { System.out.println(Q.remove()); Thread.sleep(1000); }
}
}
```

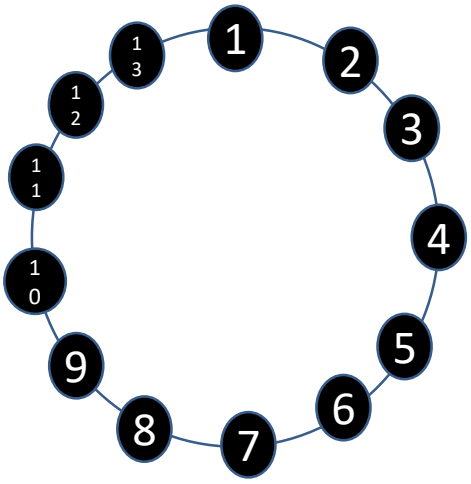
Type of Operation	Throws exception	Returns special value
Insert	`add(e)`	`offer(e)`
Remove	`remove()`	`poll()`
Examine	`element()`	`peek()`

19

Ví dụ Ứng dụng Hàng đợi [JOSEPHUS]

N= 13

K= 3



Có n người xếp thành vòng tròn, đánh số từ 1 đến n.

K là khoá.

Chọn 1 người làm vua trong n người.

Loại người ở vị trí thứ k trên vòng tròn.

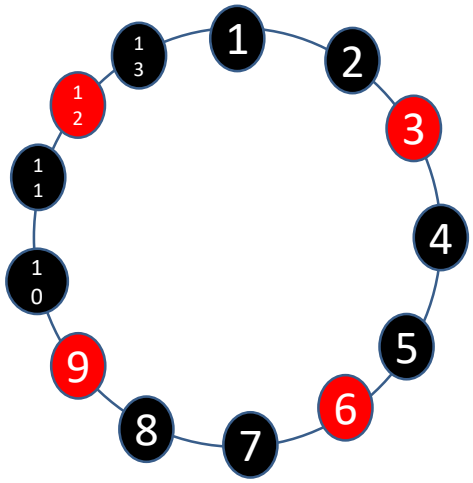
Cho biết người làm vua?

20

Ví dụ Ứng dụng Hàng đợi [JOSEPHUS]

N= 13

K= 3

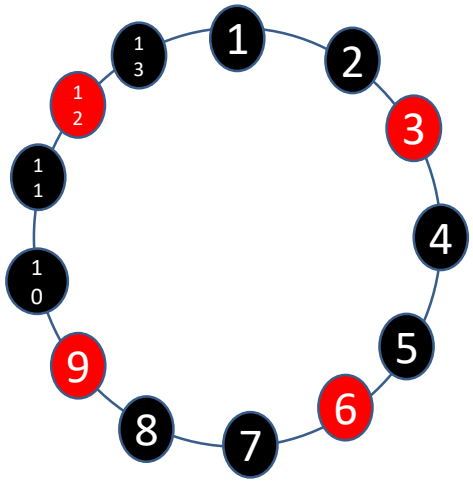


21

Ví dụ Ứng dụng Hàng đợi [JOSEPHUS]

N= 13

K= 3

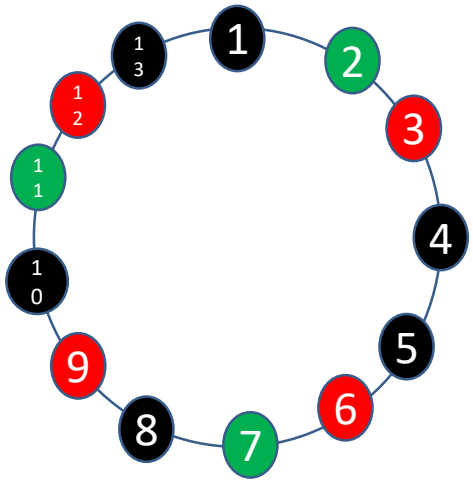


22

Ví dụ Ứng dụng Hàng đợi [JOSEPHUS]

N= 13

K= 3

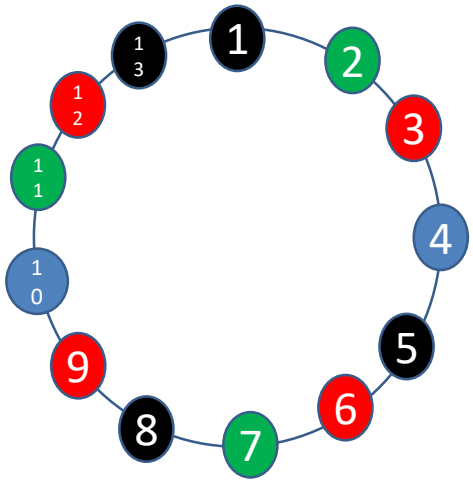


23

Ví dụ Ứng dụng Hàng đợi [JOSEPHUS]

N= 13

K= 3

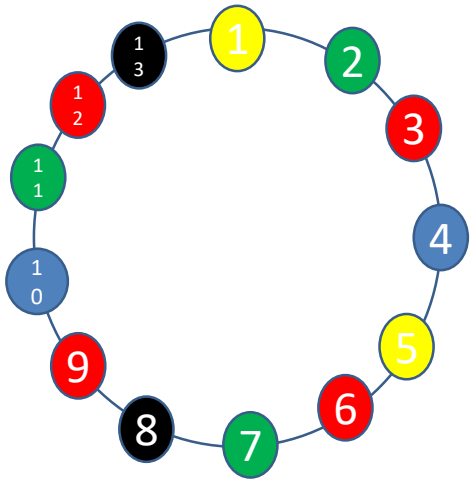


24

Ví dụ Ứng dụng Hàng đợi [JOSEPHUS]

N= 13

K= 3

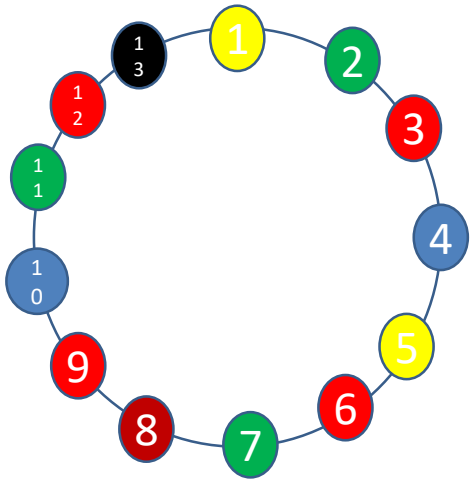


25

Ví dụ Ứng dụng Hàng đợi [JOSEPHUS]

N= 13

K= 3



26

Ví dụ Ứng dụng Hàng đợi [JOSEPHUS]

Giải pháp 1: Josephus(int n, int k)

1. Khởi tạo mảng a gồm n phần tử được đánh số từ 1 đến n
2. Dem=n; i=1, vt=0;
3. Lặp khi dem>1
   nếu a[i]>0 thì vt=vt+1;
   nếu vt%k=0 và a[i]>0 thì { a[i]=0; vt=0; dem--;}
   i=i+1;
   nếu i>n thì i=1;
4. Duyệt mảng tìm phần tử lớn hơn 0 và in ra kết quả

27

Ví dụ Ứng dụng Hàng đợi [JOSEPHUS]

Viết chương trình:

void Josephus(int n, int k) { }

Nhập n, k và thử kết quả

N=13, k=3

N=1000000, k= 5

N=1000000, k= 500

Ghi lại thời gian thực thi

28

Ví dụ Ứng dụng Hàng đợi [JOSEPHUS]

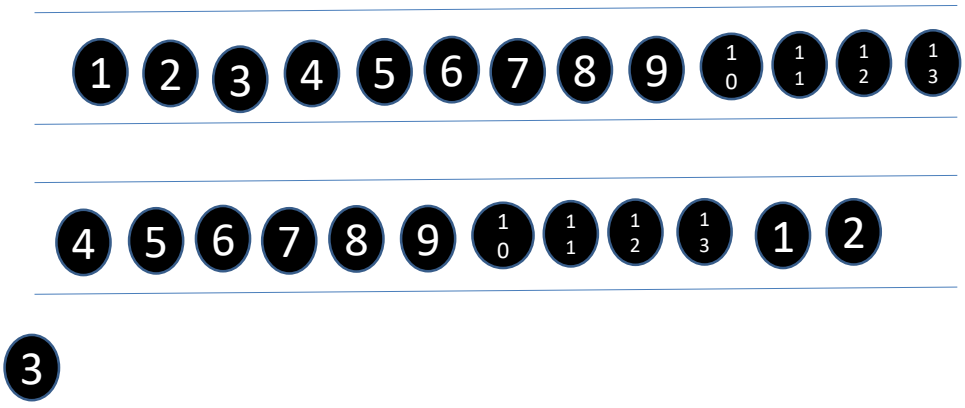
```
void Josephus(int n, int k) {
        int []a; a= new int[n+1];  for(int i=0; i<=n; i++)a[i] =i;
        int dem=n, vt=0, i=1;
          while(dem>1)
        {
                if(a[i]>0) vt++;
                if(a[i]>0 && vt%k==0) { vt=0; dem--; a[i]=0;}
                i=i+1;
                if(i>n) i=1;
        }
        S.O.P("\n Vua la: “);  for(int i=1; i<=n; i++) if(a[i]>0) { S.O.P(i + " “); break;}
}
```

29

Dùng Hàng đợi cho bài Josephus

N= 13

K= 3



30

Ví dụ Ứng dụng Hàng đợi (Queue)

N= 13

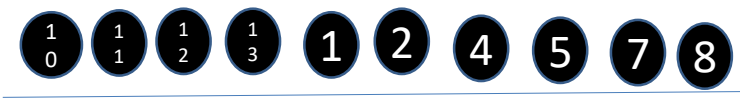
K= 3

31

Ví dụ Ứng dụng Hàng đợi (Queue)

N= 13

K= 3



32

Ví dụ Ứng dụng Hàng đợi (Queue)

N= 13

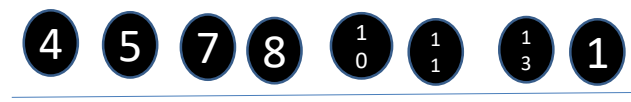
K= 3

33

Ví dụ Ứng dụng Hàng đợi (Queue)

N= 13

K= 3

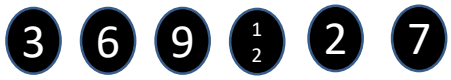


34

Ví dụ Ứng dụng Hàng đợi (Queue)

N= 13

K= 3



35

Ví dụ Ứng dụng Hàng đợi (Queue)

N= 13

K= 3

36

Ví dụ Ứng dụng Hàng đợi (Queue)

N= 13

K= 3

37

Ví dụ Ứng dụng Hàng đợi (Queue)

N= 13

K= 3

38

Ví dụ Ứng dụng Hàng đợi (Queue)

N= 13

K= 3

39

Ví dụ Ứng dụng Hàng đợi (Queue)

N= 13

K= 3

40

Ví dụ Ứng dụng Hàng đợi (Queue)

N= 13

K= 3

41

Ví dụ Ứng dụng Hàng đợi [JOSEPHUS]

Giải pháp 2: Josephus2(int n, int k)

1. Khởi tạo hàng đợi Q chứa n phần tử được đánh số từ 1 đến n
2. Dem=n; vt=0;
3. Lặp khi dem>1
   Lấy phần tử x ở đầu hàng đợi Q;
   vt=vt+1;
   nếu vt%k=0 thì { vt=0; dem--;}
   ngược lại thêm x vào sau hàng đợi Q;
4. Lấy phần tử ra khỏi Q và in ra kết quả

42

Ví dụ Ứng dụng Hàng đợi (Queue)

Cho file OPR.inp chứa n lệnh.

Lệnh $+v_i$ có nghĩa là thêm giá trị v_i vào hàng đợi

Lệnh – có nghĩa là lấy max trong hàng đợi.

Hãy thực hiện các lệnh trong file OPR.inp sau đó ghi các kết quả ra file OPR.out gồm các số còn lại trong hàng đợi theo thứ tự giảm dần

```
7
+4
+5
-
+9
-
+7
+1
```

```
7 4 1
```

43

Các thao tác Cơ bản trên Hàng đợi (Queue)

```
static <E> List<E> heapSort(Collection<E> c)
{
        Queue<E> queue = new PriorityQueue<E>(c);
        List<E> result = new ArrayList<E>();
        while (!queue.isEmpty()) result.add(queue.remove());
        return result;
}
```

44

Hàng đợi ưu tiên (Priority Queue)

```
public class MyPriorityQueue {
   Node head;
   MyPriorityQueue() { head= null;}
   boolean EmptyQ() {  return head==null;}
   int RemoveQ()    {
      int x=0;
      if(head==null) System.out.println(" \n Hang doi rong!");
      else  {      x= head.data; head= head.next;      }
      return x;
   }
```

```
class Node
{
     int data;
     Node next;
     Node(int x) {data=x; next = null;}
     Node(int x, Node t) {data=x; next = t;}
}
```

45

Hàng đợi ưu tiên (Priority Queue)

```
public class MyPriorityQueue {
   void AddQ(int x)    {
      Node t= new Node(x);
      if(head== null) head=t;
      else          if(x>=head.data) { t.next= head; head=t;}
                    else                        {
                          Node y= head,p= head;
                          while (p!=null && p.data>x)       {     y= p; p=p.next;    }
                          y.next=t; t.next=p;
                     }
   }
}
```

46

```
void xuly()   {
     try {
          FileReader fi= new FileReader("D:\\OPR.inp");    Scanner kb = new Scanner(fi);
          MyPriorityQueue Q= new MyPriorityQueue();
     int n= kb.nextInt();
     for(int i=1; i<=n; i++) {
        String t = kb.next();
        if(t.charAt(0)=='-')    { int x= Q.RemoveQ();   System.out.print("\n Lay phan tu max: " +x);       }
        else   {
    int p= Integer.parseInt(t);  System.out.print("\n Them phan tu "+ p + " vao hang doi");
           Q.AddQ(p);
        }
     }
      System.out.print("\n CAC SO CON LAI TRONG HANG DOI LA:\n");
      while (!Q.EmptyQ()) System.out.print(" " + Q.RemoveQ());
     } catch (FileNotFoundException e) {System.out.print("\n LOI DOC FILE: \n" + e.getMessage());}
   }
```

47

Bài tập về nhà

1. Cài đặt bài toán Josephus
2. Cài đặt thuật toán sắp xếp radix sort
3. Cài đặt thuật toán chuyển biểu thức trung tố sang dạng hậu tố

48

Tài liệu đọc thêm về ngăn xếp và hàng đợi

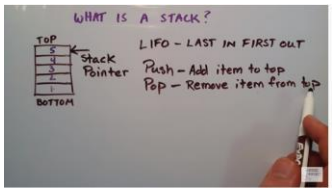
https://www.geeksforgeeks.org/queue-data-structure/

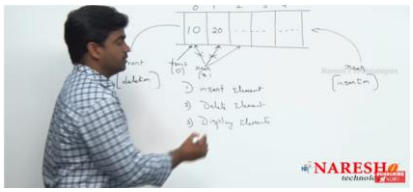
https://ocw.mit.edu/courses/civil-and-environmental-engineering/1-204-computer-algorithms-in-systems-engineering-spring-2010/lecture-notes/MIT1_204S10_lec06.pdf

49

Link YouTube

https://www.youtube.com/watch?v=FNZ5o9S9prU
https://www.youtube.com/watch?v=bxRVz8zklWM
https://www.youtube.com/watch?v=gnYM_G1ILm0





50