String :

package day14;

public class task1 {

public static void main(String args[])

{

String s1="java programming";

String s2 = "Java";

String s3 = " Java";

String s4 = "java";

System.***out***.println(s1.charAt(2)); // s1 --> v

System.***out***.println(s1.indexOf("in"));

System.***out***.println(s1.lastIndexOf("a"));

System.***out***.println(s1.replace("java" , "python"));

System.***out***.println(s1); // it print the java programming because strings are immutable

System.***out***.println(s1.substring(1,13));

System.***out***.println(s1.startsWith("ja"));

System.***out***.println(s1.endsWith("ing")); // true

System.***out***.println(s1.equals(s3)); // false

System.***out***.println(s2.equals(s3.trim())); // trim is remove the space .

System.***out***.println(s2.equals(s4)); // false

System.***out***.println(s2.equalsIgnoreCase(s4)); // true

System.***out***.println(s2.compareTo(s4));

System.***out***.println(s2.compareToIgnoreCase(s4));

}

}

v

13

10

python programming

java programming

ava programm

true

true

false

true

false

true

-32

0

package day14;

public class task2 {

public static void main(String args[])

{

String s1="Thi";

String arr[] = s1.split(" ");

for(String str : arr)

{

System.***out***.println(str);

System.***out***.println("\*\*");

}

char c\_arr[] = new char[s1.length()];

s1.getChars( 0,3,c\_arr, 0);

System.***out***.println(c\_arr);

Thi

\*\*

Thi

Since string are immutable object it is not advisable to keep on assigning the changes to the string object hence java support the two different options for mutable string object

1 . StringBuilder

2. StringBuffer

**STRINGBUILDER :**

This are mutable object . this directely manipulate the string object

Constructor :

StringBuilder str = new StringBuilder();

STRINGBUFFER :

StringBuffer str = new StringBuffer();

package day14;

public class task3 {

public static void main(String args[])

{

StringBuffer sb = new StringBuffer("java code");

sb.append("hari");

System.***out***.println(sb);

sb.insert(5,"practise");

System.***out***.println(sb);

sb.indexOf("practise");

System.***out***.println(sb.indexOf("practise"));

System.***out***.println(sb.lastIndexOf("se"));

System.***out***.println(sb.replace(0,3,"python"));

System.***out***.println(sb.substring(0));

System.***out***.println(sb.substring(0,3));

System.***out***.println(sb.reverse());

}

}

java codehari

java practisecodehari

5

11

pythona practisecodehari

pythona practisecodehari

pyt

irahedocesitcarp anohtyp

STRING BUILDER .

public class task3 {

public static void main(String args[])

{

StringBuilder sb = new StringBuilder("java code");

sb.append("hari");

System.***out***.println(sb);

sb.insert(5,"practise");

System.***out***.println(sb);

sb.indexOf("practise");

System.***out***.println(sb.indexOf("practise"));

System.***out***.println(sb.lastIndexOf("se"));

System.***out***.println(sb.replace(0,3,"python"));

System.***out***.println(sb.substring(0));

System.***out***.println(sb.substring(0,3));

System.***out***.println(sb.reverse());

}

}

java codehari

java practisecodehari

5

11

pythona practisecodehari

pythona practisecodehari

pyt

irahedocesitcarp anohtyp

public class TASK4 {

public static void main(String args[])

{

char arr[] = {'a','b','c','d','e'};

String str = String.*copyValueOf*(arr,0,5);

System.***out***.println(str);

}

}

abcde

**Capacity Of StringBuffer :**

If the min capacity obj pass as the argument is <0 or no action taken .

Current capacity of the str buffer is < argu of min capacity then new internal array is allocated for new capacity .

If the min capacity argument is > than the twise the older capacity plus 2 new capacity = min capacity . else new capacity twise the old capacity

StringBuffer sb = new StringBuffer(5); // mutable string objet

System.***out***.println(sb.capacity());

sb.ensureCapacity(8);

System.***out***.println(sb.capacity()); // capacity twise the older capacity (n\*2+2)

5

12

Queue :  
 🡪 FIFO order .

package day14;

class QueueArray {

int front, rear, size, capacity;

int arr[];

public QueueArray(int capacity) {

this.capacity = capacity;

arr = new int[capacity];

front = 0;

rear = -1;

size = 0;

}

public void Enqueue(int val) {

if (size == capacity) {

System.***out***.println("Queue Overflow");

return;

}

rear = (rear + 1) % capacity;

System.***out***.println(rear);

arr[rear] = val;

size++;

System.***out***.println(val + " enqueued to queue");

}

public int Dequeue()

{

if(size==0)

{

System.***out***.println("underflow");

}

int remove=arr[front];

front=(front+1)%capacity;

size--;

return remove;

}

public boolean isEmpty() {

if(size==0)

{

return true;

}

else {

return false;

}

}

public int peek()

{

return arr[rear];

}

public void display()

{

if(size==0)

{

System.***out***.println("empty");

}

for(int i=0;i<size;i++)

{

front=(front+i)%capacity;

System.***out***.println(arr[front]);

}

}

}

public class task6 {

public static void main(String args[])

{

QueueArray a = new QueueArray(3);

System.***out***.println("enter the value for enqueue operation :");

a.Enqueue(10);

a.Enqueue(20);

a.Enqueue(30);

System.***out***.println(a.Dequeue());

System.***out***.println("check empty : " + a.isEmpty());

System.***out***.println("peek value :"+a.peek());

a.display();

//a.Enqueue(10);

a.display();

}

}

enter the value for enqueue operation :

0

10 enqueued to queue

1

20 enqueued to queue

2

30 enqueued to queue

10

check empty : false

peek value :30

20

30

30

10

Circular queue:

(Rear-front+size)%size;

package day15;

class CircularQueue{

int front,rear,size,capacity;

int arr[];

public CircularQueue(int capacity)

{

this.capacity=capacity;

arr = new int[capacity];

front=-1;

rear = -1;

size=0;

}

public void enqueue(int val)

{

if(rear+1%capacity==front)

{

System.***out***.println("queue is full");

return ;

}

if(front==-1)

{

front=0;

}

rear=(rear+1)%capacity;

arr[rear]=val;

size++;

}

public int dequeue()

{

if(front==-1)

{

System.***out***.println("queue is empty");

}

int remove = arr[front];

if(front==rear)

{

front=rear=-1;

}

else

{

front=front+1%capacity;

}

size--;

return remove;

}

public void display()

{

int i=front;

if(front==-1)

{

System.***out***.println("queue empty...");

}

while(true)

{

if(i==rear) {

break;

}

System.***out***.println(arr[i]);

i=i%capacity;

}

}

public boolean isEmpty()

{

System.***out***.println(size);

if(size==0)

{

return true;

}

return false;

}

}

public class task1 {

public static void main(String args[])

{

CircularQueue cq = new CircularQueue(5);

cq.enqueue(10);

cq.enqueue(11);

cq.enqueue(22);

System.***out***.println(cq.dequeue());

System.***out***.println(cq.dequeue());

System.***out***.println(cq.dequeue());

//System.out.println("check empty : " + cq.isEmpty());

System.***out***.println(cq.isEmpty());

}

}

10

11

22

0

true