**Singly linked list**

1)in insertatfirst

If empty make newnode head or stor ehead in newnode next and make it as head

no return

if(head==null) {

System.out.println("empty");

return;

}

if(head==null) {

System.out.println("empty");

newNode=head;

}

////head is null so u actually making newnode value null

//aasign make newnode as head assign head value to new node

public void insertAtFirst(Student data) {

Node newNode= new Node(data);

if(newNode==null) {

return ;

}

if(head==null) {

System.out.println("empty");

head=newNode;

return ;

}

newNode.next=head;

head=newNode;

return ;

}

2)if(position==1) {

insertAtFirst(data);

return;

3)

deleteAtfirst

public void deleteAtFirst() {

if(head==null) {

return;

}

head=head.next;

}

**Doublylinkedlist**

1)create student class(dm,constr,tostring)

2)node class with (dm,constr)

3)studDatabase class

Node head;

public void insert(int data) {

public void insert(Student data) {

newNode.prev=curr;

4) in main

Scanner sc = new Scanner(System.in);

StudentDatabase sd= new StudentDatabase();

5) int ch;

do {

System.out.println("Menu:");

ch=sc.nextInt();

switch(ch) {

case 1:

default:System.out.println("invalid");

}

}while(ch!=4);

**Stack**

1)dm=(maxsize,top,arr)

2)gen constructor only maxsize

And values this.maxsize = maxsize;

this.top = -1;

this.arr = new int[maxsize];

3)public boolean isFull() {

if(top==maxsize-1) {

return true;

}

return false;

4)public boolean isEmpty() {

if(top==-1)

return true;

return false;

}

5)public boolean push(int value) {

if(!isFull()) {

arr[++top]=value; //increment index and store value

return true;

}

return false;

}

6)public int pop() {

if(!isEmpty()) {

int temp=arr[top--];//stor evalue and dercement

return temp;

}

return -1;

}

7) public int peek() {

if(!isEmpty()) {

return arr[top]; //not return top

}

return -1;

}

8)Stack s=new Stack(5)

The output `[I@5d22bbb7` you're seeing is the default `toString` representation of an array in Java. In your `display` method, you are attempting to print the entire array `arr`:

```java

public void display() {

for (int v : arr) {

System.out.println(arr + " ");

}

}

```

However, you are printing the array itself (`arr`) instead of the individual elements. This results in the default `toString` method of the array being called, which produces a string representation of the form `[I@5d22bbb7` where:

- `[I`: Indicates an array of integers.

- `@5d22bbb7`: Represents the hash code or memory address of the array.

To fix this, you should print the individual elements inside the loop:

```java

public void display() {

for (int v : arr) {

System.out.println(v + " ");

}

}

```

This way, you will print each element of the array rather than the array itself. After making this change, you should see the actual values stored in the array when you call `display`.

11) s.peek();

s.pop();

o/p 0 0

Give Sysout

private Book[] arr;

public Stack(int size) {

super();

this.size = size;

this.top = -1;

this.arr = new Book[size];

public Book pop() {

if(!isFull()) {

return arr[top--];

}

return null;

}

public void display() {

for(int i=0;i<=top;i++) {

System.out.print(arr[i]+" ");

}

System.out.println();

}

public Book search(String title) {

for(int i=0;i<=top;i++) {

if(arr[i].getTitle().equals(title)){

return arr[i];

}

}

return null;

}

2)

public void updated(String title,Book updatedBook) {

for(int i=0;i<=top;i++) {

if(arr[i].getTitle().equals(title)) {

arr[i]=updatedBook; //not return ibook then it does // // not save it

}

}

}

case 6:

s.updated(sc.next(), new Book(sc.next(),sc.next()));

3) sort

In book class public int compareTo(Book ibook) {

return this.title.compareToIgnoreCase(ibook.title);

}

public void sort() {

for(int i=0;i<=top;i++) {

for(int j=0;j<top-i;j++) {

if(arr[j].compareTo(arr[j+1])>0) {

Book temp=arr[j];

arr[j]=arr[j+1];

arr[j+1]=temp;

}

}

}

}

**Queue**

public void enqueue(int value) {

if(!isFull()) {

arr[++end]=value; //not end++ or else arrayindex out of bounds

}

}

public int dequeue() { //store->shift->decremet->return

//store front value

//shift index to left(to remove front) and update(decrement) last index value

if(!isEmpty()) {

int fvalue=arr[0];

for(int i=1;i<=end;i++) { //not 0 or else indexout of bunds

arr[i-1]=arr[i];

}

end--;

return fvalue; //this out of for loop or else it prints remaining one not // removed one

}

return -1;

}

public int front() {

if(!isEmpty()) {

return arr[0];

}

return -1;

}

**Circularqueue or deququq**

In constr

this.Rear = Size;