ISC Questions on Data Structure

Stack and Queue

Question 1 - 2019
A linear data structure enables the user to add an address from rear end and remove address from front. Define a class Diary with the following details :
Class name: Diary
Data members/instance variables:
Q[]: array to store the addresses
size: stores the maximum capacity of the array
start: to point the index of the front end
end: to point the index of the rear end
Member functions:
Diary(int max): constructor to initialize the data member size = max, start=0 and end=0
void pushadd(String n): to add the address in the diary from the rear end if possible, otherwise display the message “NO SPACE”
String popadd(): removes and returns the address from the front end of the diary if any, else returns “?????”
void show (): displays all the addresses in the diary
(a) Specify the class Diary giving details of the functions void pushadd(String) and String popadd(). Assume that the other functions have been defined. [4]
The main function and algorithm need NOT be written.
(b) Name the entity used in the above data structure arrangement. [1]
Answer:

```
(a) class Diary {
public void pushAdd(String n) {
if(Q[0].equals(""))
Q[0] = n;
else if(end + 1 < size) Q[end + 1] = n;
else
System.out.println("NO SPACE");
}
public String popadd() {
if(start -1 >= 0)
return Q[start--];
else
return "?????";
}
}
```

(b) Queue

Question 2 - 2018
A register is an entity which can hold a maximum of 100 names. The register

enables the user to add and remove names from the topmost end only.
Define a class Register with the following details:
Class name: Register
Data members/instance variables:
stud[]: array to store the names of the students
cap: stores the maximum capacity of the array to point the index of the top end
top: to point the index of the top end
Member functions:
Register (int max) : constructor to initialize the data member cap = max, top = -1 and create the string array
void push(String n): to add names in the register at the top location if possible, otherwise display the message “OVERFLOW” String pop(): removes and returns the names from the topmost location of the register if any, else returns “$$”
void display (): displays all the names in the register
(a) Specify the class Register giving details of the functions void push(String) and String pop().
Assume that the other functions have been defined. [4]
The main function and algorithm need NOT be written.
(b) Name the entity used in the above data structure arrangement. [1]

Question 3- 2017
A queue is an entity which can hold a maximum of 100 integers. The queue enables the user to add integers from the rear and remove integers from the front. [5]
Define a class Queue with the following details:
Class name: Queue
Data members/instance variables:
Que[]: array to hold the integer elements
size: stores the size of the array
front: to point the index of the front
rear: to point the index of the rear
Member functions:
Queue(int mm): constructor to initialize the data
size = mm, front = 0, rear = 0
void addele(int v): to add integer from the rear if possible else display the message “Overflow”
int delele(): returns elements from front if present, otherwise displays the message “Underflow” and return-9999
void display(): displays the array elements
Specify the class Queue giving details of ONLY the functions void addele(int) and int delete()
Assume that the other functions have been defined.
The main function and algorithm need NOT be written.

Question 4 -2016
A bookshelf is designed to store the books in a stack with LIFO(Last In First Out) operation. Define a class Book with the following specifications: [5]
Class name: Book

Data members/instance variables:
name[]: stores the names of the books
point: stores the index of the topmost book
max: stores the maximum capacity of the bookshelf
Methods/Member functions:
Book(int cap): constructor to initialise the data members
max = cap and point = -1
void tell(): displays the name of the book which was last entered in the shelf. If there is no book left in the shelf, displays the message “SHELF EMPTY”
void add(String v): adds the name of the book to the shelf if possible, otherwise displays the message ‘SHELF FULL”
void display(): displays all the names of the books available on the shelf
Specify the class Book giving the details of ONLY the functions void tell() and void add(String). Assume that the other functions have been defined.
The main function need not be written.

Question 5- 2015
WordPile is an entity which can hold a maximum of 20 characters. The restriction is that a character can be added or removed from one end only.
Some of the members of classes are given below:
Class name: WordPile
Data members/instance variables:
ch[]: character array to hold the character elements
capacity: integer variable to store the maximum capacity
top: to point to the index of the topmost element
Member functions/methods:
WordPile (int cap): constructor to initialise the data member
capacity = cap, top = -1 and create the WordPile
void pushChar(char v): adds the character to the top of WordPile if possible, otherwise output a message “WordPile is full”
charpopChar(): returns the deleted character from the top of the WordPile if possible, otherwise it returns ‘\\’
(a) Specify the class WordPile giving the details of the constructor, void pushChar(char) and char popChar(). [8]
The main function and algorithm need not be written.
(b) What is the name of the entity described above and state one of its applications? [2]

Question 6- 2014
A stack is a linear data structure which enables the user to add and remove integers from one end only, using the concept of LIFO(Last In First Out). An array containing the marks of 50 students in ascending order is to be pushed into the stack.
Define a class Array_to_Stack with the following details: [10]
Class name: Array to Stack
Data members/instance variables:
m[]: to store the marks
st[ ]: to store the stack elements

cap: maximum capacity of the array and stack
top: to point the index of the topmost element of the stack
Methods/Member functions:
Array_to_Stack(int n): parameterized constructor to initialize cap = n and top = -1
void input_marks(): to input the marks from the user and store it in the array m[ ] in ascending order and simultaneously push the marks into the stack st[ ] by invoking the function pushmarks()
void pushmarks(int v): to push the marks into the stack at seetop location if possible, otherwise, display “not possible”
intpopmarks(): to return marks from the stack if possible, otherwise, return-999
void display(): To display the stack elements
Specify the class Array_to_Stack, giving the details of the constructor(int), void input_marks(), void pushmarks(int), int popmarks() and void display().
The main function and the algorithm need not be written.

Question 7 – 2013 D Queue
A doubly queue is a linear data structure which enables the user to add and remove integers from either ends, i.e. from front or rear. Define a class Dequeue with the following details: [10]
Class name: Dequeue
Data members/instance variables:
arr[ ]: array to hold up to 100 integer elements
lim: stores the limit of the dequeue
front: to point to the index of the front end
rear: to point to the index of the rear end
Member functions:
Dequeue(int 1): constructor to initialize the data members lim = 1; front = rear = 0
void addfront(int val): to add integer from the front if possible else display the message (“Overflow from front”) voidaddrear(intval): to add integer from the rear if possible else display the message (“Overflow from rear”)
int popfront(): returns element from front, if possible otherwise returns – 9999
int poprear(): returns element from rear, if possible otherwise returns – 9999
Specify the class Dequeue giving details of the constructor (int), void addfront(int), void addrear (int, popfront ( ) and int poprear ( ). The main function and algorithm need not be written.

Question 8-2012
Link is an entity which can hold a maximum of 100 integers. Link enables the user to add elements from the rear end and remove integers from the front end of the entity. Define a class Link with the following details:
Class name: Link
Data Members/instance variables:
Ink []: entity to hold the integer elements,
max: stores the maximum capacity of the entity,
begin: to point to the index of the front end.
end: to point to the index of the rear end.
Member functions:

Link(intmm): constructor to initialize max = mm. begin = 0. end = 0.
void addlink (int v): to add an element from the rear index if possible otherwise display the message “OUT OF SIZE… ”
int dellink(): to remove and return an element from the front index. if possible otherwise display the message “EMPTY …” and return – 99.
void display(): displays the elements of the entity.
(a) Specify the class Link giving details of the constructor (int), void addlink (int), int dellink() and void display ( ). [9]
THE MAIN FUNCTION AND ALGORITHM NEED NOT BE WRITTEN.
(b) What type of data structure is the above entity? [1]
Answer:

```
(a) class Link
{
int Ink [ ] = new int [100];
int begin, end. max;
public Link (int mm)
{
max = mm;
begin = end = 0;
}
void addlink(int v)
{
if (end==0)
{
end = begin = 1;
JiInk [end] = v;
}
else if (end == max)
{
System.out.println("List is Full");
}
else
Ink [++end] = v;
}
}
int dellink ()
{
int a;
if (begin == 0)
{
System.out.println(" Empty...");
return - 99;
}
else if (begin == end)
{
a = Ink [begin];
```

```
begin = end = 0;
return a;
}
else
{
a = Ink [begin];
begin++;
return (a);
}
}
void display()
{
int i;
for(i = begin; i < = end; i++)
{
System.out.println(Ink[i]);
}
}
}
```

(b) It is a Queue.

Question 9 -2011

A stack is a kind of data structure which can store elements with the restriction that an element can be added or removed from the top only.

The details of the class Stack is given below:

Class name: Stack

Data members/instance variables:

st[]: the array to hold the names

size: the maximum capacity of the string array

top: the index of the topmost element of the stack

ctr: to count the number of elements of the stack

Member functions:

Stack( ): default constructor

Stack(int cap): constructor to initialize size = cap and top = -1

void pushname(String n): to push a name onto the stack. If the stack is full, display the message “OVERFLOW”

String popname(): removes a name from the top of the stack and returns it. If the stack is empty, display the message “UNDERFLOW”

void display(): Display the elements of the stack.

(a) Specify class Stack giving details of the constructors(), void pushname(String n), String popname() and void display(). [8]

THE MAIN() FUNCTION AND ALGORITHM NEED NOT BE WRITTEN.

(b) Under what Principle does the above entity work