import java.util.Scanner;

class Stack {

private int arr[];

private int top;

private int capacity;

Stack(int size) {

if (size <= 0) {

System.out.println("Invalid size; creating stack of size 10 by default.");

size = 10;

}

this.capacity = size;

this.arr = new int[capacity];

this.top = -1; // empty

}

Stack(int[] inputArr) {

if (inputArr == null) {

this.capacity = 10;

this.arr = new int[capacity];

this.top = -1;

return;

}

this.capacity = inputArr.length;

this.arr = new int[capacity];

for (int i = 0; i < inputArr.length; i++) {

arr[i] = inputArr[i];

}

this.top = inputArr.length - 1;

}

void push(int x) {

if (top == capacity - 1) {

System.out.println("Stack overflow! Cannot push " + x + ".");

return;

}

arr[++top] = x;

System.out.println("Pushed " + x + " to stack.");

}

void push(int x, int y) {

int free = capacity - (top + 1);

if (free <= 0) {

System.out.println("Stack overflow! Cannot push " + x + " and " + y + ".");

return;

}

else if (free == 1) {

push(x);

System.out.println("Only one element could be pushed; " + y + " not pushed due to limited space.");

}

else {

push(x);

push(y);

}

}

int pop() {

if (top == -1) {

System.out.println("Stack underflow! Nothing to pop.");

return Integer.MIN\_VALUE;

}

int val = arr[top--];

System.out.println("Popped " + val + " from stack.");

return val;

}

void pop(int n) {

if (n <= 0) {

System.out.println("Enter a positive number of elements to pop.");

return;

}

if (top == -1) {

System.out.println("Stack underflow! No elements to pop.");

return;

}

System.out.println("Popping up to " + n + " element(s):");

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

if (top == -1) {

System.out.println("No more elements to pop.");

break;

}

int val = arr[top--];

System.out.println(" Popped: " + val);

}

}

void display() {

if (top == -1) {

System.out.println("Stack is empty.");

return;

}

System.out.println("Stack (top -> bottom):");

for (int i = top; i >= 0; i--) {

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

}

}

void display(int n) {

if (n <= 0) {

System.out.println("Enter a positive number of elements to display.");

return;

}

if (top == -1) {

System.out.println("Stack is empty.");

return;

}

System.out.println("Top " + n + " element(s) (top -> ...):");

int count = 0;

for (int i = top; i >= 0 && count < n; i--, count++) {

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

}

if (count < n) {

System.out.println(" (Only " + count + " element(s) present in stack.)");

}

}

}

public class LibraryStackDemo {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

Stack st = null;

System.out.println("Choose how to initialize stack:");

System.out.println("1. Empty stack with size");

System.out.println("2. Stack initialized with array");

System.out.print("Enter choice (1 or 2): ");

int choice = sc.nextInt();

if (choice == 1) {

System.out.print("Enter size of stack: ");

int size = sc.nextInt();

st = new Stack(size);

} else if (choice == 2) {

System.out.print("Enter number of elements: ");

int n = sc.nextInt();

if (n < 0) n = 0;

int arr[] = new int[n];

if (n > 0) {

System.out.println("Enter elements (bottom -> top order): ");

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

arr[i] = sc.nextInt();

}

}

st = new Stack(arr);

}

else {

System.out.println("Invalid choice. Creating an empty stack of size 10 by default.");

st = new Stack(10);

}

int option;

do {

System.out.println("\n--- Stack Menu ---");

System.out.println("1. Push one element");

System.out.println("2. Push two elements");

System.out.println("3. Pop one element");

System.out.println("4. Pop multiple elements");

System.out.println("5. Display full stack");

System.out.println("6. Display top n elements");

System.out.println("7. Exit");

System.out.print("Enter your choice: ");

option = sc.nextInt();

switch(option) {

case 1:

System.out.print("Enter element: ");

int val = sc.nextInt();

st.push(val);

break;

case 2:

System.out.print("Enter two elements (separated by space): ");

int v1 = sc.nextInt();

int v2 = sc.nextInt();

st.push(v1, v2);

break;

case 3:

st.pop();

break;

case 4:

System.out.print("Enter number of elements to pop: ");

int numToPop = sc.nextInt();

st.pop(numToPop);

break;

case 5:

st.display();

break;

case 6:

System.out.print("Enter number of top elements: ");

int topN = sc.nextInt();

st.display(topN);

break;

case 7:

System.out.println("Exiting... Goodbye!");

break;

default:

System.out.println("Invalid choice! Try again.");

}

}

while(option != 7);

sc.close();

}

}

OUTPUT:

Choose how to initialize stack

1. Empty stack with size

2. Stack initialized with array

2

Enter number of elements: 3

Enter elements:

4

5

6

Stack Menu

1. Push one element

2. Push two elements

3. Pop one element

4. Pop multiple elements

5. Display full stack

6. Display top n elements

7. Exit

Enter your choice: 3

Popped: 6

Stack Menu

1. Push one element

2. Push two elements

3. Pop one element

4. Pop multiple elements

5. Display full stack

6. Display top n elements

7. Exit

Enter your choice: 1

Enter element: 7

Pushed: 7

Stack Menu

1. Push one element

2. Push two elements

3. Pop one element

4. Pop multiple elements

5. Display full stack

6. Display top n elements

7. Exit

Enter your choice: 5

Stack contents (Top to Bottom):

7

5

4

Stack Menu

1. Push one element

2. Push two elements

3. Pop one element

4. Pop multiple elements

5. Display full stack

6. Display top n elements

7. Exit

Enter your choice: 6

Enter number of top elements: 2

Top 2 elements:

7

5

Stack Menu

1. Push one element

2. Push two elements

3. Pop one element

4. Pop multiple elements

5. Display full stack

6. Display top n elements

7. Exit

Enter your choice: 7

Exiting