**Lab 9: Develop a stack class to hold a maximum of 10 integers with suitable methods. Develop a JAVA main method to illustrate Stack operations.**

import java.util.Scanner;

public class Stack {

private static final int MAX\_SIZE = 10;

private int[] stackArray;

private int top;

public Stack() {

stackArray = new int[MAX\_SIZE];

top = -1;

}

public void push(int value) {

if (top < MAX\_SIZE - 1) {

stackArray[++top] = value;

System.out.println("Pushed: " + value);

} else {

System.out.println("Stack Overflow! Cannot push " + value + ".");

}

}

public int pop() {

if (top >= 0) {

int poppedValue = stackArray[top--];

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

return poppedValue;

} else {

System.out.println("Stack Underflow! Cannot pop from an empty stack.");

return -1; // Return a default value for simplicity

}

}

public int peek() {

if (top >= 0) {

System.out.println("Peeked: " + stackArray[top]);

return stackArray[top];

} else {

System.out.println("Stack is empty. Cannot peek.");

return -1; // Return a default value for simplicity

}

}

public void display() {

if (top >= 0) {

System.out.print("Stack Contents: ");

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

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

}

System.out.println();

} else {

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

}

}

public boolean isEmpty() {

return top == -1;

}

public boolean isFull() {

return top == MAX\_SIZE - 1;

}

public static void main(String[] args) {

Stack stack = new Stack();

Scanner scanner = new Scanner(System.in);

int choice;

do {

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

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

System.out.println("2. Pop");

System.out.println("3. Peek");

System.out.println("4. Display Stack Contents");

System.out.println("5. Check if the stack is empty");

System.out.println("6. Check if the stack is full");

System.out.println("0. Exit");

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

choice = scanner.nextInt();

switch (choice) {

case 1:

System.out.print("Enter the value to push: ");

int valueToPush = scanner.nextInt();

stack.push(valueToPush);

break;

case 2:

stack.pop();

break;

case 3:

stack.peek();

break;

case 4:

stack.display();

break;

case 5:

System.out.println("Is the stack empty? " + stack.isEmpty());

break;

case 6:

System.out.println("Is the stack full? " + stack.isFull());

break;

case 0:

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

break;

default:

System.out.println("Invalid choice. Please try again.");

}

} while (choice != 0);

scanner.close();

}

}

**Output**

$ java Stack

Stack Menu:

1. Push

2. Pop

3. Peek

4. Display Stack Contents

5. Check if the stack is empty

6. Check if the stack is full

0. Exit

Enter your choice: 4

Stack is empty.

Stack Menu:

1. Push

2. Pop

3. Peek

4. Display Stack Contents

5. Check if the stack is empty

6. Check if the stack is full

0. Exit

Enter your choice: 5

Is the stack empty? true

Stack Menu:

1. Push

2. Pop

3. Peek

4. Display Stack Contents

5. Check if the stack is empty

6. Check if the stack is full

0. Exit

Enter your choice: 6

Is the stack full? false

Stack Menu:

1. Push

2. Pop

3. Peek

4. Display Stack Contents

5. Check if the stack is empty

6. Check if the stack is full

0. Exit

Enter your choice: 1

Enter the value to push: 10

Pushed: 10

Stack Menu:

1. Push

2. Pop

3. Peek

4. Display Stack Contents

5. Check if the stack is empty

6. Check if the stack is full

0. Exit

Enter your choice: 1

Enter the value to push: 20

Pushed: 20

Stack Menu:

1. Push

2. Pop

3. Peek

4. Display Stack Contents

5. Check if the stack is empty

6. Check if the stack is full

0. Exit

Enter your choice: 4

Stack Contents: 10 20

Stack Menu:

1. Push

2. Pop

3. Peek

4. Display Stack Contents

5. Check if the stack is empty

6. Check if the stack is full

0. Exit

Enter your choice: 3

Peeked: 20

Stack Menu:

1. Push

2. Pop

3. Peek

4. Display Stack Contents

5. Check if the stack is empty

6. Check if the stack is full

0. Exit

Enter your choice: 1

Enter the value to push: 30

Pushed: 30

Stack Menu:

1. Push

2. Pop

3. Peek

4. Display Stack Contents

5. Check if the stack is empty

6. Check if the stack is full

0. Exit

Enter your choice: 4

Stack Contents: 10 20 30

Stack Menu:

1. Push

2. Pop

3. Peek

4. Display Stack Contents

5. Check if the stack is empty

6. Check if the stack is full

0. Exit

Enter your choice: 2

Popped: 30

Stack Menu:

1. Push

2. Pop

3. Peek

4. Display Stack Contents

5. Check if the stack is empty

6. Check if the stack is full

0. Exit

Enter your choice: 3

Peeked: 20

Stack Menu:

1. Push

2. Pop

3. Peek

4. Display Stack Contents

5. Check if the stack is empty

6. Check if the stack is full

0. Exit

Enter your choice: 4

Stack Contents: 10 20

Stack Menu:

1. Push

2. Pop

3. Peek

4. Display Stack Contents

5. Check if the stack is empty

6. Check if the stack is full

0. Exit

Enter your choice: 0

Exiting the program. Goodbye!