Practice3: implement with array

import java.util.Arrays;

import java.util.Scanner;

class ArrayStackDemo {

public static void main(String[] args) {

ArrayStack a = new ArrayStack(5);

boolean flag = true;// 用于判断循环结束的标志

Scanner sc = new Scanner(System.in);

String key = "";// 用于接受菜单的选项

while (flag) {

System.out.println("show：显示栈");

System.out.println("exit：退出程序");

System.out.println("push：进栈");

System.out.println("pop：出栈");

key = sc.nextLine();

switch (key) {

case "show":

a.show();

break;

case "exit":

flag = false;

System.out.println("程序结束！");

break;

case "push":

System.out.println("请输入要进栈的数据：");

int val = sc.nextInt();

a.push(val);

break;

case "pop":

int pop = a.pop();

System.out.println("出栈的值是：" + pop);

}

}

}

}

class ArrayStack {

private int MaxSize;// 定义数组的最大长度

private int[] arr;// 定义数组，数据就放在该数组

private int top = -1;// 定义栈顶，初始化数据为-1

public ArrayStack(int maxSize) {

this.MaxSize = maxSize;

arr = new int[MaxSize];

}

// 判断数组是否为空

public boolean isEmpty() {

return top == -1;

}

// 判断数组是否满了

public boolean isFull() {

System.out.println("栈顶：" + top + "最大长度：" + MaxSize);

return top == MaxSize -1;

}

// 进栈

public void push(int val) {

// 先判断栈是否满了，满了就不能添加进去

if (isFull()) {

System.out.println("栈已经满了~~");

return;

}

top++;

arr[top] = val;

}

// 出栈

public int pop() {

// 先判断栈是否为空

if (isEmpty()) {

throw new RuntimeException("栈为空，无法出栈！");

}

int val = arr[top];

top--;

return val;

}

public void show() {

if (isEmpty()) {

System.out.println("没有数据");

return;

}

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

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

}

System.out.println();

}

}