**(1.1)VALUECHECKER:**

**package** valuechecker;

**import** java.util.Scanner;

**public** **class** ValueChecker {

**public** **static** **void** main(String[] args) {

Scanner in = **new** Scanner(System.***in***);

**int** value = 0;

System.***out***.println("Enter a number:");

value = in.nextInt();

**if**(value == 7 ) {

System.***out***.println("That's lucky");

}

**else** **if**(value == 13) {

System.***out***.println("That's unlucky!");

}

**else** {

System.***out***.println("That number is neither lucky nor unlucky!");

}

in.close();

}

}

**(1.1)INPUT VARIABLES:**

**package** inputvariables;

**import** java.util.Scanner;

**public** **class** InputVariables {

**public** **static** **void** main(String[] args) {

Scanner in = **new** Scanner(System.***in***);

**boolean** boolval;

**byte** byteval;

**char** charval;

**short** shortval;

**int** intval;

**long** longval;

**float** floatval;

**double** doubleval;

System.***out***.print("Please enter a boolean value:");

boolval = in.nextBoolean();

System.***out***.print("Please enter a byte value:");

byteval = in.nextByte();

System.***out***.print("Please enter a char value:");

charval = in.next().charAt(0);

System.***out***.print("Please enter a short value:");

shortval = in.nextShort();

System.***out***.print("Please enter an int value:");

intval = in.nextInt();

System.***out***.print("Please enter a long value:");

longval = in.nextLong();

System.***out***.print("Please enter a float value:");

floatval = in.nextFloat();

System.***out***.print("Please enter a double value:");

doubleval = in.nextDouble();

in.close();

System.***out***.println("boolean value: " + boolval);

System.***out***.println("byte value : " + byteval);

System.***out***.println("char value : " + charval);

System.***out***.println("short value : " + shortval);

System.***out***.println("int value : " + intval);

System.***out***.println("long value : " + longval);

System.***out***.println("float value : " + floatval);

System.***out***.println("double value : " + doubleval);

}

}

**(2.1)TESTBANK:**

**package** testbank;

**public** **class** TestBank {

**public** **static** **void** main(String[] args) {

Account a1 = **new** Account("Sanjay Gupta", 11556, 300);

Account a2 = **new** Account();

Account a3 = **new** Account();

a2.setAccountName("He Xai");

a2.setAccountNum(22338);

a2.setBalance(500);

a3.setAccountName("Ilya Mustafana");

a3.setAccountNum(44559);

a3.setBalance(1000);

a1.print();

a2.print();

a3.print();

}

}

**class** Account {

**private** String accountName;

**private** **int** accountNum;

**private** **double** balance;

**public** Account(String accountName, **int** accountNum, **double** balance) {

**this**.accountName = accountName;

**this**.accountNum = accountNum;

**this**.balance = balance;

}

**public** Account() {

}

**public** **void** setAccountName(String accountName) {

**this**.accountName = accountName;

}

**public** **void** setAccountNum(**int** accountNum) {

**this**.accountNum = accountNum;

}

**public** **void** setBalance(**double** balance) {

**this**.balance = balance;

}

**public** **void** print() {

System.***out***.println("Account Name: " + accountName);

System.***out***.println("Account Number: " + accountNum);

System.***out***.println("Balance: " + balance);

System.***out***.println();

}

}

**(2.2)BAKERYDRIVER:**

**package** bakery;

**public** **class** BakeryDriver {

**public** **static** **void** main(String[] args) {

BakeryItem baguette = **new** Bread("Baguette", 2.50, "Wheat");

BakeryItem chocolateCake = **new** Cake("Chocolate Cake", 15.00, "Chocolate");

System.***out***.println(baguette);

System.***out***.println(chocolateCake);

}

}

**(2.3)INTERFACEBANKACCOUNT:**

**package** bankaccount;

**public** **interface** InterfaceBankAccount {

**public** **final** String ***BANK*** = "JavaBank";

**public** **void** deposit(**int** amt);

**public** **void** withdraw(**int** amt);

**public** **int** getBalance();

**public** String getBankName();

}

**(2.3)ABSTRACTBANKACCOUNT:**

package abstractbankaccount;

abstract class AbstractBankAccount {

public final String BANK = "JavaBank";

abstract public void deposit(int amt);

abstract public void withdraw(int amt);

abstract public int getBalance();

abstract public String getBankName();

}

**(2.4)ASSERTION:**

**package** assertions;

**import** java.util.Scanner;

**public** **class** AssertEx {

**public** **static** **void** main(String[] args) {

Scanner in = **new** Scanner(System.***in***);

**try** {

System.***out***.print("Please enter a number: ");

**double** x = in.nextDouble();

System.***out***.println("Value of x: " + x);

**int** y = (**int**) x;

System.***out***.println("Value of y (after casting to int): " + y);

// Inform user about potential data loss

**if** (x != y) {

System.***out***.println("Warning: Casting double to int results in loss of fractional part.");

}

} **catch** (Exception e) {

System.***out***.println("Invalid input! Please enter a valid number.");

} **finally** {

in.close();

}

}

**GENERICS:** }

**(3.1)CELL:**

**package** cell;

**public** **class** Cell {

**private** Object data;

**public** **void** setValue(Object cellData) {

data = cellData;

}

**public** Object getValue() {

**return** data;

}

}

**(3.1)CELLDRIVER:**

**package** celldriver;

**public** **class** CellDrive {

**public** **static** **void** main(String[] args) {

Cell<Integer, String> mixCell = **new** Cell<Integer, String>();

Cell<Integer, Integer> integerCell = **new** Cell<>();

mixCell.setValue(1, "4");

integerCell.setValue(45, 60);

**int** mcType1 = mixCell.getT1Value();

String mcType2 = mixCell.getT2Value();

**int** icType1 = integerCell.getT1Value();

**int** icType2 = integerCell.getT2Value();

System.***out***.println(mixCell);

System.***out***.println(integerCell);

System.***out***.println("The numerical value is: " + mcType1 + ". The text value is: " + mcType2);

System.***out***.println("The first numerical value is: " + icType1 + " and the second is: " + icType2);

}

}

**class** Cell<T1, T2> {

**private** T1 value1;

**private** T2 value2;

**public** **void** setValue(T1 value1, T2 value2) {

**this**.value1 = value1;

**this**.value2 = value2;

}

**public** T1 getT1Value() {

**return** value1;

}

**public** T2 getT2Value() {

**return** value2;

}

@Override

**public** String toString() {

**return** "Cell{" +

"value1=" + value1 +

", value2=" + value2 +

'}';

}

}

**(3.1)GENERICMETHODDRIVER:**

**package** genericmethoddriver;

**public** **class** GenericMethodDriver {

**public** **static** **void** main(String[] args) {

GenericMethodClass gmc = **new** GenericMethodClass();

Integer[] integerArray = {1, 2, 3};

String[] stringArray = {"This", "is", "fun"};

gmc.printArray(integerArray);

gmc.printArray(stringArray);

}

}

**class** GenericMethodClass {

**public** <T> **void** printArray(T[] array) {

**for** (T element : array) {

System.***out***.print(element + " ");

}

System.***out***.println();

}

}

**(3.2)COLLECTIONS(STUDENT):**

**package** student1;

**import** java.util.ArrayList;

**import** java.util.Collections;

**public** **class** Student {

**public** **static** **void** main(String[] args) {

ArrayList<String> studentNames = **new** ArrayList<>(); // Corrected ArrayList instantiation

*addStudents*(studentNames); // Add students to the list

*displayStudents*(studentNames); // Display the list of students

Collections.*sort*(studentNames); // Sort the list of students

*displayStudents*(studentNames); // Display the sorted list of students

}

**static** **void** displayStudents(ArrayList<String> studentNames) {

**for** (String student : studentNames) {

System.***out***.println("Student Name: " + student); // Corrected the output statement

}

}

**static** **void** addStudents(ArrayList<String> studentNames) {

studentNames.add("Alice");

studentNames.add("Bob");

studentNames.add("Charlie");

studentNames.add("Diana");

studentNames.add("Eve");

}

}

**(3.3)COLLECTIONS:**

**HASHMAP:**

import java.util.HashMap;

public class HashMapExample {

public static void main(String[] args) {

HashMap<String, String> fruitBowl = new HashMap<String, String>();

fruitBowl.put("Apple", "Red");

fruitBowl.put("Banana", "Yellow");

fruitBowl.put("Grape", "Purple");

fruitBowl.put("Orange", "Orange");

fruitBowl.put("Lemon", "Yellow");

for (String fruit : fruitBowl.keySet()) {

System.out.println(fruit + " is " + fruitBowl.get(fruit));

}

}

}

**(3.3)**

**LINKED LIST QUEUE:**

**package** linkedlistqueue;

**import** java.util.LinkedList;

**public** **class** LettersQueue {

**public** **static** **void** main(String[] args) {

LinkedList<String> lettersQ = **new** LinkedList<String>();

lettersQ.add("A");

lettersQ.add("B");

lettersQ.add("C");

lettersQ.add("D");

lettersQ.add("E");

System.***out***.println("Queue: " + lettersQ);

System.***out***.println("Removed: " + lettersQ.poll()); // Removes "A"

System.***out***.println("Removed: " + lettersQ.poll()); // Removes "B"

System.***out***.println("Queue after removals: " + lettersQ);

System.***out***.println("Peek at the first element: " + lettersQ.peek());

System.***out***.println("Does the queue contain 'C'? " + lettersQ.contains("C"));

System.***out***.println("Does the queue contain 'A'? " + lettersQ.contains("A"));

}

}

**(3.3)**

**LINKED LIST STACK:**

**package** linkedliststack;

**import** java.util.LinkedList;

**public** **class** LettersStack {

**public** **static** **void** main(String[] args) {

LinkedList<String> letterS = **new** LinkedList<String>();

letterS.push("A");

letterS.push("B");

letterS.push("C");

letterS.push("D");

letterS.push("E");

System.***out***.println("Stack: " + letterS);

System.***out***.println("Popped: " + letterS.pop()); // Removes "E"

System.***out***.println("Popped: " + letterS.pop()); // Removes "D"

System.***out***.println("Stack after pops: " + letterS);

System.***out***.println("Peek at the top element: " + letterS.peek());

System.***out***.println("Does the stack contain 'C'? " + letterS.contains("C"));

System.***out***.println("Does the stack contain 'E'? " + letterS.contains("E"));

}

}

**(3.3)**

**SORTING A COLLECTIONS:**

**package** collection;

**public** **class** Student {

**private** String firstName;

**private** String lastName;

**private** **int** mark;

**public** Student(String firstName, String lastName, **int** mark) {

**this**.firstName = firstName;

**this**.lastName = lastName;

**this**.mark = mark;

}

**public** String getFirstName() {

**return** firstName;

}

**public** **void** setFirstName(String firstName) {

**this**.firstName = firstName;

}

**public** String getLastName() {

**return** lastName;

}

**public** **void** setLastName(String lastName) {

**this**.lastName = lastName;

}

**public** **int** getMark() {

**return** mark;

}

**public** **void** setMark(**int** mark) {

**this**.mark = mark;

}

**public** String toString() {

**return** "Student [First Name: " + firstName + ", Last Name: " + lastName + ", Mark: " + mark + "]";

}

**public** **static** **void** main(String[] args) {

Student student = **new** Student("John", "Doe", 85);

System.***out***.println(student);

student.setFirstName("Jane");

student.setLastName("Smith");

student.setMark(90);

System.***out***.println(student);

}

}

**(3.4)**

**SELECTION SORT:**

package selectionsort;

public class SortExample {

public static void main(String[] args) {

int[] numbers = {40, 7, 59, 4, 1};

System.*out*.println("Array before sorting:");

*displayValues*(numbers);

*selectionSort*(numbers);

System.*out*.println("Array after sorting:");

*displayValues*(numbers);

}

static void selectionSort(int[] numbers) {

int n = numbers.length;

for (int i = 0; i < n - 1; i++) {

int minIndex = i;

for (int j = i + 1; j < n; j++) {

if (numbers[j] < numbers[minIndex]) {

minIndex = j; // Update minIndex if a smaller element is found

}

}

int temp = numbers[minIndex];

numbers[minIndex] = numbers[i];

numbers[i] = temp;

}

}

static void displayValues(int[] numbers) {

for (int number : numbers) {

System.*out*.print(number + " ");

}

System.*out*.println(); // Move to the next line after displaying the array

}

}

**(3.4)**

**BUBBLE SORT:**

**package** bubblesort;

**public** **class** SortExample {

**public** **static** **void** main(String[] args) {

**int**[] numbers = {40, 7, 59, 4, 1};

System.***out***.println("Array before sorting:");

*displayValues*(numbers);

*bubbleSort*(numbers);

System.***out***.println("Array after sorting:");

*displayValues*(numbers);

}

**static** **void** bubbleSort(**int**[] numbers) {

**int** n = numbers.length;

**boolean** swapped;

**for** (**int** i = 0; i < n - 1; i++) {

swapped = **false**;

**for** (**int** j = 0; j < n - 1 - i; j++) {

**if** (numbers[j] > numbers[j + 1]) {

**int** temp = numbers[j];

numbers[j] = numbers[j + 1];

numbers[j + 1] = temp;

swapped = **true**;

}

}

**if** (!swapped) **break**;

}

}

**static** **void** displayValues(**int**[] numbers) {

**for** (**int** number : numbers) {

System.***out***.print(number + " ");

}

System.***out***.println(); // Move to the next line after displaying the array

}

}

**(3.4)**

**SEQUENTIAL SEARCH:**

**package** sequentialsearch;

**public** **class** SortExample {

**public** **static** **void** main(String[] args) {

**int**[] numbers = {40, 7, 59, 4, 1};

System.***out***.println("Array before sorting:");

*displayValues*(numbers);

*sequentialSearch*(numbers, 13);

*sequentialSearch*(numbers, 7);

*bubbleSort*(numbers);

System.***out***.println("Array after sorting:");

*displayValues*(numbers);

}

**static** **void** sequentialSearch(**int**[] numbers, **int** target) {

**boolean** found = **false**;

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

**if** (numbers[i] == target) {

System.***out***.println("Value " + target + " found at index " + i);

found = **true**;

**break**;

}

}

**if** (!found) {

System.***out***.println("Value " + target + " not found in the array.");

}

}

**static** **void** bubbleSort(**int**[] numbers) {

**int** n = numbers.length;

**boolean** swapped;

**for** (**int** i = 0; i < n - 1; i++) {

swapped = **false**;

**for** (**int** j = 0; j < n - 1 - i; j++) {

**if** (numbers[j] > numbers[j + 1]) {

// Swap numbers[j] and numbers[j + 1]

**int** temp = numbers[j];

numbers[j] = numbers[j + 1];

numbers[j + 1] = temp;

swapped = **true**;

}

}

**if** (!swapped) **break**;

}

}

**static** **void** displayValues(**int**[] numbers) {

**for** (**int** number : numbers) {

System.***out***.print(number + " ");

}

System.***out***.println(); // Move to the next line after displaying the array

}

}