Question 1:

* Primitive data types are the most fundamental built-in data types that a programming language offers. They are directly kept in memory, have a fixed size, and are compared or assigned based on their actual values. On the other hand, reference data types are user-defined types that are allocated or compared according to their references. They might have a range of sizes and are stored as memory locations.

Question 2:

* The area of a program where a variable can be accessed and utilized is known as its scope. While global variables can be accessed from anywhere in the program, local variables are restricted to the block or function in which they are declared.

Question 3:

* To avoid unpredictable actions, stop mistakes, guarantee reliable program execution, and give default values, variables must be initialized.

Question 4:

* Instance variables belong to a specific instance of a class, while local variables are restricted to a certain block or method. Static variables are shared by all instances of a class.

Question 5:

* Widening casting automatically converts smaller data types to bigger data types, whereas narrowing casting involves intentional casting and may cause data loss when converting larger data types to smaller data types.

Question 6:

- boolean:

- Size: 1 bit

- Default: false

- Range: true or false (boolean values)

- char:

- Size: 2 bytes

- Default: ‘\u0000’ (null character)

- Range: ‘\u0000’ to ‘\uffff’ (Unicode characters)

- byte:

- Size: 1 byte

- Default: 0

- Range: -128 to 127

- short:

- Size: 2 bytes

- Default: 0

- Range: -32,768 to 32,767

- int:

- Size: 4 bytes

- Default: 0

- Range: -2,147,483,648 to 2,147,483,647

- long:

- Size: 8 bytes

- Default: 0L

- Range: -9,223,372,036,854,775,808 to 9,223,372,036,854,775,807

- float:

- Size: 4 bytes

- Default: 0.0f

- Range: 1.4E-45 to 3.4028235E+38

- double:

- Size: 8 bytes

- Default: 0.0

- Range: 4.9E-324 to 1.7976931348623157E+308

Question 7:

* A package is a mechanism to bundle similar classes, interfaces, and sub-packages together in Java programming. It offers namespace management, access control, and modular organization for code entities.

Question 8:

* Java packages are essential because they offer a method for managing and organizing code, enforcing encapsulation and access control, preventing naming conflicts, promoting code sharing and reuse, enabling import and compilation control, and promoting developer collaboration.

SECTION 2

Question 1:

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

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

String surname = scanner.nextLine();

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

int age = scanner.nextInt();

int surnameLength = surname.length();

String ageType = (age % 2 == 0) ? "even" : "odd";

System.out.println("The number of characters in your surname is " + surnameLength + ".");

System.out.println("Your current age is an " + ageType + " number.");

scanner.close();

}

}

The Scanner class is used in this software to read user input from the console. This user is first prompted for their surname, which is then stored in the surname variable. After that, we ask them to provide their age which is kept in the age variable.

Next, we use the length() method to compute the surname’s character count and save the result in the surnameLength variable. Using the modulus operator%, we may establish whether an age is even or odd by seeing if the residue after age is divided by two is 0. If so, the age is even; if not, it is strange. The ageType variable is used to hold the outcome.

Finally, we print the number of characters in the surname and whether the age is even or odd.

Question 2:

Import java.util.Scanner;

Public class AverageCalculator {

Public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

Int totalUnits = 5;

Int[] marks = new int[totalUnits];

System.out.println(“Enter the marks for each unit:”);

For (int I = 0; I < totalUnits; i++) {

System.out.print(“Unit “ + (I + 1) + “: “);

Marks[i] = scanner.nextInt();

}

Double average = calculateAverage(marks);

System.out.printf(“Average marks: %.2f%n”, average);

}

Public static double calculateAverage(int[] marks) {

Int sum = 0;

For (int mark : marks) {

Sum += mark;

}

Return (double) sum / marks.length;

}

}

In this software, the marks for each unit are stored in an array called marks. Using a loop, the user is asked to input the marks for each unit. The average is determined by adding together all the marks and dividing by the total number of units using the calculateAverage technique.

The average is shown with two decimal places using the System.out.printf statement and the %.2f format specifier.

The pupil will be prompted to enter their grades for each unit they did and after you compile and run the software. The software will compute the average after receiving the input and display it on the screen.

Question 3:

Import java.util.Scanner;

Public class DivisibilityTest {

Public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.print(“Enter an integer: “);

Int number = scanner.nextInt();

Boolean divisible = false;

For (int I = 0; I <= 9; i++) {

If (number % I == 0) {

Divisible = true;

System.out.println(“The number is divisible by “ + i);

}

}

If (!divisible) {

System.out.println(“The number is not divisible by any integer from 0 to 9”);

}

}

}

The user of this program is asked to enter an integer. Following that, a loop iterates through the integers 0 through 9. The modulo operator% is used to determine for each number in this range if the supplied integer is divisible by that number. If the condition is satisfied, a message saying that the number is divisible by that specific integer is printed.

The given number is not divisible by any integer from 0 to 9 if the divisible flag is still false after the loop, and a message stating this is printed.

When you compile and run the program, it will prompt you for an integer. Following input, the software will examine each number in the range for divisibility.

Question 4:

public class MultiplesFinder {

public static void main(String[] args) {

int startRange = 71; // Starting range

int endRange = 150; // Ending range

System.out.println("Multiples of 2, 3, and 7 within the range " + startRange + " to " + endRange + ":");

for (int i = startRange; i <= endRange; i++) {

if (i % 2 == 0 || i % 3 == 0 || i % 7 == 0) {

System.out.println(i); // Display the multiple

}

}

}

}

In this program, the start range and finish range are both set to 71 and 150 respectively. The modulo operator% is then used to iteratively go through each number in this rangeimport java.util.Scanner;

Public class Calculator {

Public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

// Prompt the user to enter the first number

System.out.print(“Enter the first number: “);

Double num1 = scanner.nextDouble();

// Prompt the user to enter the operation (+, -, \*, /)

System.out.print(“Enter the operation (+, -, \*, /): “);

Char operator = scanner.next().charAt(0);

// Prompt the user to enter the second number

System.out.print(“Enter the second number: “);

Double num2 = scanner.nextDouble();

Double result;

// Perform the operation based on the operator selected

Switch (operator) {

Case ‘+’:

// Addition operation

Result = num1 + num2;

System.out.println(“Result: “ + result);

Break;

Case ‘-‘:

// Subtraction operation

Result = num1 – num2;

System.out.println(“Result: “ + result);

Break;

Case ‘\*’:

// Multiplication operation

Result = num1 \* num2;

System.out.println(“Result: “ + result);

Break;

Case ‘/’:

// Division operation

If (num2 != 0) {

Result = num1 / num2;

System.out.println(“Result: “ + result);

} else {

// Display error message if dividing by zero

System.out.println(“Error: Cannot divide by zero”);

}

Break;

Default:

// Display error message for invalid operator

System.out.println(“Error: Invalid operator”);

}

}

}

and determine whether it is a multiple of 2, 3, or 7. The number is printed if the condition is true.

When you compile and run the program, all the multiples of 2, 3, and 7 between 71 and 150 will be shown.

Question 5:

import java.util.Scanner;

public class Calculator {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

// Prompt the user to enter the first number

System.out.print("Enter the first number: ");

double num1 = scanner.nextDouble();

// Prompt the user to enter the operation (+, -, \*, /)

System.out.print("Enter the operation (+, -, \*, /): ");

char operator = scanner.next().charAt(0);

// Prompt the user to enter the second number

System.out.print("Enter the second number: ");

double num2 = scanner.nextDouble();

double result;

// Perform the operation based on the operator selected

switch (operator) {

case '+':

// Addition operation

result = num1 + num2;

System.out.println("Result: " + result);

break;

case '-':

// Subtraction operation

result = num1 - num2;

System.out.println("Result: " + result);

break;

case '\*':

// Multiplication operation

result = num1 \* num2;

System.out.println("Result: " + result);

break;

case '/':

// Division operation

if (num2 != 0) {

result = num1 / num2;

System.out.println("Result: " + result);

} else {

// Display error message if dividing by zero

System.out.println("Error: Cannot divide by zero");

}

break;

default:

// Display error message for invalid operator

System.out.println("Error: Invalid operator");

}

}

}

A prompt from the application asks the user to enter two numbers and choose from the operations (+, -, \*, or /). The appropriate calculation is then carried out, and the outcome is displayed, using a switch statement. Incorrect division by zero is also handled by the program. Once the software has run, the user can enter the numbers and the operation, and the outcome will be shown on the screen.