If all of a program’s methods are static, the program is **procedural**

The data variables in each object are known as **fields** or **attributes**

A **class** is the template / blueprint / pattern / mold / “cookie cutter” from which objects are created

Leave off the static keyword for OOP methods

|  |  |
| --- | --- |
| Rectangle | |
| - length : double  - width : double | |
| + setLength(len : double) : void  + setWidth(w : double) : void  + getLength() : double  + getWidth() : double  + getArea() : double | |
| **Package** | **Description** | |
| java.applet | Classes necessary to create an applet | |
| java.awt | Classes for the Abstract Windowing Toolkit used in drawing and GUIs | |
| java.io | Classes for various types of input and output | |
| java.lang | General classes (automatically imported) | |
| java.net | Classes for network communications | |
| java.security | Classes that implement security features | |
| java.sql | Classes for accessing databases | |
| java.text | Various classes for formatting text | |
| java.util | Various utility classes | |
| javax.swing | Classes for creating GUIs | |

import java.util.Random;

public class Die

{

private int sides; // Number of sides

private int value; // The die's value

public Die(int numSides)

{

sides = numSides;

roll();

}

public void roll()

{

Random rand = new Random();

value = rand.nextInt(sides) + 1;

}

public int getSides()

{

return sides;

}

public int getValue()

{

return value;

}

}

public class DieArgument

{

public static void main(String[] args)

{

final int SIX\_SIDES = 6;

final int TWENTY\_SIDES = 20;

Die sixDie = new Die(SIX\_SIDES);

Die twentyDie = new Die(TWENTY\_SIDES);

rollDie(sixDie);

rollDie(twentyDie);

}

public static void rollDie(Die d)

{

System.out.println("Rolling a " + d.getSides() +

" sided die.");

d.roll();

System.out.println("The die's value: " + d.getValue());

}

}

An **array** is a set of variables declared with the same name and type

public class PassElements

{

public static void main(String[] args)

{

int[] numbers = {5, 10, 15, 20, 25, 30, 35, 40};

for (int index = 0; index < numbers.length; index++)

showValue(numbers[index]);

}

public static void showValue(int n)

{

System.out.print(n + " ");

}

}

import java.util.Scanner; // Needed for Scanner class

public class PassArray

{

public static void main(String[] args)

{

final int ARRAY\_SIZE = 4; // Size of the array

int[] numbers = new int[ARRAY\_SIZE];

getValues(numbers);

System.out.println("Here are the " +

"numbers that you entered:");

showArray(numbers);

}

private static void getValues(int[] array)

{

// Create a Scanner objects for keyboard input.

Scanner keyboard = new Scanner(System.in);

System.out.println("Enter a series of " +

array.length + " numbers.");

for (int index = 0; index < array.length; index++)

{

System.out.print("Enter number " +

(index + 1) + ": ");

array[index] = keyboard.nextInt();

}

}

public static void showArray(int[] array)

{

for (int index = 0; index < array.length; index++)

System.out.print(array[index] + " ");

}

}

import java.util.Scanner;

public class BinarySearchDemo

{

public static void main(String[] args)

{

int numbers[] = {101, 142, 147, 189, 199, 207, 222,

234, 289, 296, 310, 319, 388, 394,

417, 429, 447, 521, 536, 600};

int result, searchValue;

String input;

Scanner keyboard = new Scanner(System.in);

do

{

System.out.print("Enter a value to search for: ");

searchValue = keyboard.nextInt();

result = binarySearch(numbers, searchValue);

if (result == -1)

System.out.println(searchValue + " was not found.");

else

{

System.out.println(searchValue + " was found at " +

"element " + result);

}

keyboard.nextLine();

System.out.print("Do you want to search again? (Y or N): ");

input = keyboard.nextLine();

} while (input.charAt(0) == 'y' || input.charAt(0) == 'Y');

}

public static int binarySearch(int[] array, int value)

{

int first; // First array element

int last; // Last array element

int middle; // Midpoint of search

int position; // Position of search value

boolean found; // Flag

first = 0;

last = array.length - 1;

position = -1;

found = false;

while (!found && first <= last)

{

middle = (first + last) / 2;

if (array[middle] == value)

{

found = true;

position = middle;

}

else if (array[middle] > value)

last = middle - 1;

else

first = middle + 1;

}

return position;

}

}

public class SearchArray

{

public static void main(String[] args)

{

int[] tests = { 87, 75, 98, 100, 82 };

int results;

results = sequentialSearch(tests, 100);

if (results == -1)

{

System.out.println("You did not " +

"earn 100 on any test.");

}

else

{

System.out.println("You earned 100 " +

"on test " + (results + 1));

}

}

public static int sequentialSearch(int[] array,

int value)

{

int index; // Loop control variable

int element; // Element the value is found at

boolean found; // Flag indicating search results

index = 0;

element = -1;

found = false;

while (!found && index < array.length)

{

if (array[index] == value)

{

found = true;

element = index;

}

index++;

}

return element;

}

}

public class SalesData

{

private double[] sales; // The sales data

public SalesData(double[] s)

{

sales = new double[s.length];

for (int index = 0; index < s.length; index++)

sales[index] = s[index];

}

public double getTotal()

{

double total = 0.0; // Accumulator

for (int index = 0; index < sales.length; index++)

total += sales[index];

return total;

}

public double getAverage()

{

return getTotal() / sales.length;

}

public double getHighest()

{

double highest = sales[0];

for (int index = 1; index < sales.length; index++)

{

if (sales[index] > highest)

highest = sales[index];

}

return highest;

}

public double getLowest()

{

double lowest = sales[0];

for (int index = 1; index < sales.length; index++)

{

if (sales[index] < lowest)

lowest = sales[index];

}

return lowest;

}

}

import java.util.Scanner;

import java.io.\*;

public class FileArray

{

public static void main(String[] args) throws IOException

{

final int ARRAY\_SIZE = 5;

int numbers[] = new int[ARRAY\_SIZE];

int count = 0;

File file = new File("Values.txt");

if (!file.exists())

{

System.out.println("ERROR! Cannot open file.");

System.exit(0);

}

Scanner inputFile = new Scanner(file);

while ((inputFile.hasNext()) && (count < numbers.length))

{

numbers[count] = inputFile.nextInt();

count++;

}

System.out.print("These are the values read from the file: ");

for (int index = 0; index < count; index++)

{

System.out.print(numbers[index] + " ");

}

System.out.println();

}

}

import java.util.Scanner;

public class CorpSales

{

public static void main(String[] args)

{

final int DIVS = 3; // Three divisions in the company

final int QTRS = 4; // Four quarters

double totalSales = 0.0; // Accumulator

double[][] sales = new double[DIVS][QTRS];

Scanner keyboard = new Scanner(System.in);

System.out.println("This program will calculate the " +

"total sales of");

System.out.println("all the company's divisions. " +

"Enter the following sales data:");

for (int div = 0; div < DIVS; div++)

{

for (int qtr = 0; qtr < QTRS; qtr++)

{

System.out.printf("Division %d, Quarter %d: $",

(div + 1), (qtr + 1));

sales[div][qtr] = keyboard.nextDouble();

}

System.out.println(); // Print blank line.

}

for (int div = 0; div < DIVS; div++)

{

for (int qtr = 0; qtr < QTRS; qtr++)

{

totalSales += sales[div][qtr];

}

}

System.out.printf("Total company sales: $%,.2f\n",

totalSales);

}

}