

## JavaInte.java

```
1
2 import java.util.Scanner;
3
4 public class JavaInte {
5
6     public static void main(String[] args) {
7         Scanner in = new Scanner(System.in);
8         boolean quit = false;
9
10        /**
11         * @author hsilva4495
12         *
13         */
14
15        // handle user commands
16        int menuItem;
17        /**
18         * A loop statement allows us to execute a statement or group of statements multiple times
19        and
20         * following is the general form of a loop statement in most of the programming languages
21         *
22         * this do-while loop controls the whole of the program by endlessly looping the
23        parameters
24         * unless the boolean returns false aka user types quit menu option the methods after main
25         * includes for loops to count min and sum of arrays
26         *
27         * while loop Repeats a statement or group of statements while a given condition is true.
28        It
29         * tests the condition before executing the loop body.
30         *
31         * for loop Execute a sequence of statements multiple times and abbreviates the code that
32         * manages the loop variable.
33         *
34         * do...while loop Like a while statement, except that it tests the condition at the end
35        of the
36         * loop body.
37         */
38        do {
39            try {
40                System.out.println("\n\n");
41                System.out.println("Hello Welcome to my Program!!");
42                System.out.println("Please choose an item from the menu to continue.\n");
43                System.out.println("(type in a number as indicated)");
44                System.out.println("\t1. Variable");
45                System.out.println("\t2. Java data Types");
46                System.out.println("\t3. Mathematical operations");
47                System.out.println("\t4. Integer division");
48                System.out.println("\t5. if and else statements");
49                System.out.println("\t6. Conditional and relational operators");
50                System.out.println("\t7. array");
51                System.out.println("\t8. setters and getters example");
52                System.out.println("\t9. 2D array");
53                System.out.println("\t10. Quit");
54
55                System.out.println("\nPlease Choose a main menu item:");
56
57                menuItem = in.nextInt();
```

```

54
55     switch (menuItem) {
56
57         case 1:
58             System.out.println("Variables:");
59             System.out.println("\tA variable is a value that can change,\n "
60                 + "depending on conditions or on information passed to the program. ");
61
62             in.nextLine();
63             System.out.println("Press Enter to go to Start");
64             in.nextLine();
65
66             break;
67
68         case 2:
69
70             dataType();
71
72             in.nextLine();
73             System.out.println("Press Enter to go to Start");
74             in.nextLine();
75
76             break;
77
78         case 3:
79
80             addmultOpoperations();
81
82             in.nextLine();
83             System.out.println("Press Enter to go to Start");
84             in.nextLine();
85
86             break;
87
88         case 4:
89
90             intDivision();
91
92             in.nextLine();
93             System.out.println("Press Enter to go to Start");
94             in.nextLine();
95
96             break;
97
98         case 5:
99             System.out.println("if and else statements");
100             System.out.println(
101                 "The if-then statement is the most basic of all the control flow statements. "
102                 + "\nIt tells your program to execute a certain section of code only if "
103                 + "a particular test evaluates to true.");
104             System.out.println("\nFor Example, type in a number to see if it's even.");
105
106             int ThisIsAChoice = in.nextInt();
107
108             ifElseStatement(ThisIsAChoice);
109
110             in.nextLine();

```

```

111         System.out.println("Press Enter to go to Start");
112         in.nextLine();
113
114         break;
115
116     case 6:
117
118         // conditional menu
119
120         int condi_menu;
121
122         System.out.println("\nPlease select an option: \n");
123         System.out.println("1. Relational operators");
124         System.out.println("2. Conditional operators");
125
126         condi_menu = in.nextInt();
127
128         switch (condi_menu) {
129             case 1:
130
131                 System.out.println(" \t relational operator is a programming language "
132                     + "construct or operator that tests or defines some kind "
133                     + "of relation between two values.\n");
134                 System.out.println("\t In more simplistic terms, this operation "
135                     + "compares and distinguishes objects of the same type");
136                 System.out.println(" \n\tThe equality and relational operators determine if
one "
137                     + "operand is greater than, less than, equal or not equal to another
operand.");
138
139                 /*
140                 * Some of the more common symbols include == equal to != not equal to >
greater
141                 * than >= greater than or equal to < less than <= less than or equal to
142                 */
143
144                 System.out.println(
145                     "\tfor example please type in two integers in order to compare them:
\n");
146
147                 int Cond_val1 = in.nextInt();
148                 int Cond_val2 = in.nextInt();
149                 if (Cond_val1 == Cond_val2) {
150                     System.out.println("value 1 == value 2");
151                 }
152                 if (Cond_val1 != Cond_val2) {
153                     System.out.println("value1 != value2");
154                 }
155                 if (Cond_val1 > Cond_val2) {
156                     System.out.println("value1 > value2");
157                 }
158                 if (Cond_val1 < Cond_val2) {
159                     System.out.println("value1 < value2");
160                 }
161                 if (Cond_val1 <= Cond_val2) {
162                     System.out.println("value1 <= value2");
163                 }

```

```

164
165         in.nextLine();
166         System.out.println("Press Enter to go to Start");
167         in.nextLine();
168
169         break;
170
171     case 2:
172         System.out
173             .println("The && and || operators perform Conditional-AND and
Conditional-OR "
174                 + "operations on two boolean expressions. "
175                 + "\nThese operators exhibit 'short-circuiting' behavior, "
176                 + "which means that the second operand is evaluated only if needed.");
177
178         int value1 = 1;
179         int value2 = 2;
180         if ((value1 == 1) && (value2 == 2)) {
181             System.out.println("value1 is 1 AND value2 is 2");
182         }
183         if ((value1 == 1) || (value2 == 1)) {
184             System.out.println("value1 is 1 OR value2 is 1");
185         }
186
187         in.nextLine();
188         System.out.println("Press Enter to go to Start");
189         in.nextLine();
190
191         break;
192
193     default:
194         System.out.println("Invalid Choice\n back to main.");
195     }
196
197     in.nextLine();
198     System.out.println("Press Enter to go to Start");
199     in.nextLine();
200
201     break;
202
203 case 7:
204     int array_menu;
205
206     System.out.println("\nPlease select an array example: \n");
207     System.out.println("1. find smallest value in array");
208     System.out.println("2. find sum values of an array ");
209
210     array_menu = in.nextInt();
211
212     switch (array_menu) {
213     case 1:
214
215         int[] Minarray = new int[] { 10, 11, 88, 2, 12, 120 };
216         System.out.println("\nThe array values are: 10, 11, 88, 2, 12, 120");
217         // Calling MinVal() method for getting min value
218         int min = MinVal(Minarray);
219         System.out.println("Minimum Value is: " + min);

```

```

220
221         break;
222
223     case 2:
224         int[] sum_arr = new int[] { 21, 16, 86, 21, 3 };
225         System.out.println("\nThe array values are: 21,16,86,21,3");
226         int sums_fer_bums = sumVal(sum_arr);
227         System.out.println("The sum is " + sums_fer_bums);
228
229         break;
230
231     default:
232         System.out.println("Invalid Choice\n back to main.");
233     }
234
235     in.nextLine();
236     System.out.println("Press Enter to go to Start");
237     in.nextLine();
238
239     break;
240
241 case 8:
242
243     setUndget();
244
245     in.nextLine();
246     System.out.println("Press Enter to go to Start");
247     in.nextLine();
248
249     break;
250
251 case 9:
252
253     arrTwo();
254
255     in.nextLine();
256     System.out.println("Press Enter to go to Start");
257     in.nextLine();
258
259     break;
260
261 case 10:
262
263     in.nextLine();
264     System.out.println("Press Enter to go to Start");
265     in.nextLine();
266
267     break;
268
269 case 11:
270     System.out.println("\nBye-bye!");
271     quit = true;
272     break;
273
274 default:
275     System.out.println("Invalid choice.");
276

```

```

277     }
278
279     } catch (IndexOutOfBoundsException e) {
280         System.out.println("Please input a Number as told by the menu\n");
281     }
282
283     } while (!quit);
284
285     System.out.println("Hope to see you later!");
286     // this closes the scanner
287
288     in.close();
289
290 }
291
292 public static void setUndget() {
293     Student tyler = new Student("Tyler"); // creating an instance of the Student class
294     Student derek = new Student("Derek");
295
296     tyler.setScore(1, 100);
297     // creating an array capable of storing objects created from the Student class
298     Student[] classroom = new Student[2];
299     classroom[0] = tyler;
300     classroom[1] = derek;
301
302     for (Student aStudent : classroom) {
303         // for each loop to go through array holding Student objects
304         System.out.println(aStudent.getName());
305         double[] studentScores = aStudent.getScores();
306         for (double score : studentScores) {
307             // for each loop to go through the array that is a field of the Student objects
308             System.out.println(score);
309         }
310     }
311 }
312
313 public static void searchArr() {
314     // this searches the array box for a value other than 0
315
316 }
317
318 public static void addmultOpperations() {
319     System.out.println("Mathematical operations");
320     System.out.println("java is capable of performing simple operation without"
321         + "\n\tthe need for a different class to be called beforehand");
322     System.out.println("for example simple arithmetic like addition, subtraction"
323         + ", \n\tmultiplication and division with int, long, float, and double " + "data
324         types.");
325     System.out.println(
326         "\nFor example we will add the two integers 5 and 2 together to demonstrate
327         addition");
328     int firstnum = 5;
329     int secondnum = 2;
330     firstnum += secondnum;
331     System.out.println("The total of the equation is: " + firstnum);
332     System.out.println(
333         "\nwe will now multiply two double (.5 and 8.00) inputs together to demonstrate

```

```

multiplication");
332     double thirtdnum = 8.00;
333     double fourthnum = 0.50;
334     fourthnum *= thirtdnum;
335     System.out.println("The total of the equation is: " + fourthnum);
336 }
337
338 public static void intDivision() {
339     System.out.println("Integer division");
340     System.out.println("java is capable of performing simple operation without"
341         + "\n\tthe need for a different class to be called beforehand");
342     System.out.println("for example simple arithmetic like addition, subtraction"
343         + ", \n\tmultiplication and division with int, long, float, and double " + "data
types.");
344     System.out.println("\nThis example will demonstrate division of 10 by the ingeter 2: ");
345     int divide = 10;
346     divide /= 2;
347     System.out.println("The total of the equation is: " + divide);
348 }
349
350 /**
351  * This is primed to demonstrate the error handling from user inputs
352  *
353  */
354 public static void ifelseStatement(int thisIsAChoice) {
355     int ifChoice = thisIsAChoice;
356     if (ifChoice % 2 == 0) {
357         System.out.println(" this number is even");
358     } else {
359         System.out.println(" this number is odd.");
360     }
361 }
362
363 /**
364  * byte: Thebytedata type is an 8-bit signed two's complement integer. It has a minimum
value of
365  * -128 and a maximum value of 127 (inclusive). short: Theshortdata type is a 16-bit signed
two's
366  * complement integer. It has a minimum value of -32,768 and a maximum value of 32,767
367  * (inclusive). int: By default, theintdata type is a 32-bit signed two's complement
integer,
368  * which has a minimum value of -231and a maximum value of 231-1. You can use the int data
type to
369  * represent an unsigned 32-bit integer, which has a minimum value of 0 and a maximum value
of
370  * 232-1. long: Thelongdata type is a 64-bit two's complement integer. The signed long has a
371  * minimum value of -263and a maximum value of 263-1.You can use the long data type to
represent
372  * an unsigned 64-bit long, which has a value of 0 to 264-1. float: As with the
recommendations
373  * for byte and short, use afloat(instead of double) if you need to save memory in large

```

```

arrays of
380 * floating point numbers. This data type should never be used for precise values, such as
381 * currency. double: For decimal values, this data type is generally the default choice. As
382 * mentioned above, this data type should never be used for precise values, such as
currency.
383 * boolean: Theboolean data type has only two possible values:trueandfalse. char: Thechar data
type
384 * is a single 16-bit Unicode character (i.e. letters and other symbols). It has a minimum
value
385 * of '\u0000'(or 0) and a maximum value of '\uffff'(or 65,535 inclusive).
386 *
387 */
388
389 public static void dataType() {
390
391     System.out.println("Java data Types:byte, short, int, long, float, double, boolean,
char");
392     System.out.println("\tFor a more detailed look at the data types"
393         + "please look into the comments of the program");
394
395 }
396
397 /**
398 *
399 * @param Minimum
400 *     this for loop counts all the array values and compares them to each other to get
min
401 * @return
402 *
403 */
404 public static int MinVal(int[] Minimum) {
405     int MinValue = Minimum[0];
406     // this for loop counts all the array values and compares them to each other to get min
407     for (int Min_count = 1; Min_count < Minimum.length; Min_count++) {
408         if (Minimum[Min_count] < MinValue) {
409             MinValue = Minimum[Min_count];
410         }
411     }
412     return MinValue;
413 }
414
415 /**
416 *
417 * @param Summations
418 *     this loop looks into the sum of all values in the array and adds them to the
counter.
419 * @return
420 *
421 */
422 public static int sumVal(int[] Summations) {
423     int sum_total = 0;
424     // this loop looks into the sum of all values in the array and adds them to the counter.
425     for (int counter_arr = 0; counter_arr < Summations.length; counter_arr++) {
426         sum_total += Summations[counter_arr];
427     }
428     return sum_total;
429 }

```



```
430 public static void arrTwo() {
431
432     // declaring and initializing 2D array
433     int arr[][] = { { 1, 2, 3 }, { 4, 5, 6 }, { 7, 8, 9 } };
434
435     // printing 2D array in the form of a table.
436     for (int i = 0; i < 3; i++) {
437         for (int j = 0; j < 3; j++) {
438             System.out.print(arr[i][j] + " ");
439         }
440         System.out.println();
441     }
442 }
443
444 }
```

## Student.java

```
1
2 public class Student {
3
4     private String name;
5     private double[] scores; // an array as the field of a class
6
7     public Student(String n) { // constructor method, called automatic when new objects are
        created
8         name = n;
9         // actually create the space in memory for the array
10        scores = new double[3];
11    }
12
13
14    public void setScore(int testNum, int score) {
15        if (testNum >= 1 && testNum <= 3) {
16            scores[testNum - 1] = score;
17        }
18    }
19
20    public String getName() {
21        return name;
22    }
23
24    public double[] getScores() {
25        return scores;
26    }
27 }
```