```java

import java.util.Scanner;

public class JavaInte {

  public static void main(String[] args) {
    Scanner in = new Scanner(System.in);
    boolean quit = false;

    /**
     * @author hsilva4495
     *
     */

    // handle user commands
    int menuItem;
    /**
     * A loop statement allows us to execute a statement or group of statements multiple times and
     * following is the general form of a loop statement in most of the programming languages
     *
     * this do-while loop controlls the whole of the program by endlessly looping the parameters
     * unless the boolean returns false aka user types quit menu option the methods after main
     * includes for loops to count min and sum of arrays
     *
     * while loop Repeats a statement or group of statements while a given condition is true. It
     * tests the condition before executing the loop body.
     *
     * for loop Execute a sequence of statements multiple times and abbreviates the code that
     * manages the loop variable.
     *
     * do...while loop Like a while statement, except that it tests the condition at the end of the
     * loop body.
     */
    do {
      try {
        System.out.println("\n\n");
        System.out.println("Hello Welcome to my Program!!");
        System.out.println("Please choose an item from the menu to continue.\n");
        System.out.println("(type in a number as indicated)");
        System.out.println("\t1. Variable");
        System.out.println("\t2. Java data Types");
        System.out.println("\t3. Mathematical operations");
        System.out.println("\t4. Integer division");
        System.out.println("\t5. if and else statements");
        System.out.println("\t6. Conditional and relational operators");
        System.out.println("\t7. array");
        System.out.println("\t8. setters and getters example");
        System.out.println("\t9. 2D array");
        System.out.println("\t10. Quit");

        System.out.println("\nPlease Choose a main menu item:");

        menuItem = in.nextInt();

        switch (menuItem) {

          case 1:
            System.out.println("Variables:");
            System.out.println("\tA variable is a value that can change,\n "
                + "depending on conditions or on information passed to the program. ");

            in.nextLine();
            System.out.println("Press Enter to go to Start");
            in.nextLine();

            break;

          case 2:

            dataType();

            in.nextLine();
            System.out.println("Press Enter to go to Start");
            in.nextLine();

            break;

          case 3:

            addmultOpperations();

            in.nextLine();
            System.out.println("Press Enter to go to Start");
            in.nextLine();

            break;

          case 4:

            intDivision();

            in.nextLine();
            System.out.println("Press Enter to go to Start");
            in.nextLine();

            break;

          case 5:
            System.out.println("if and else statements");
            System.out.println(
                "The if-then statement is the most basic of all the control flow statements. "
                    + "\nIt tells your program to execute a certain section of code only if "
                    + "a particular test evaluates to true.");
            System.out.println("\nFor Example, type in a number to see if it's even.");

            int ThisIsAChoice = in.nextInt();

            ifelseStatement(ThisIsAChoice);

            in.nextLine();
            System.out.println("Press Enter to go to Start");
            in.nextLine();

            break;

          case 6:

            // conditional menu

            int condi_menu;

            System.out.println("\nPlease select an option: \n");
            System.out.println("1. Relational operators");
            System.out.println("2. Conditional operators");

            condi_menu = in.nextInt();

            switch (condi_menu) {
              case 1:

                System.out.println(" \t relational operator is a programming language "
                    + "construct or operator that tests or defines some kind "
                    + "of relation between two values.\n");
                System.out.println("\t In more simplistic terms, this operation "
                    + "compares and distinguishes objects of the same type");
                System.out.println(" \n\tThe equality and relational operators determine if one "
                    + "operand is greater than, less than, equal or not equal to another operand.");

                /*
                 * Some of the more common symbols include == equal to != not equal to > greater
                 * than >= greater than or equal to < less than <= less than or equal to
                 */

                System.out.println(
                    "\tfor example please type in two integers in order to compare them: \n");

                int Cond_val1 = in.nextInt();
                int Cond_val2 = in.nextInt();
                if (Cond_val1 == Cond_val2) {
                  System.out.println("value 1 == value 2");
                }
                if (Cond_val1 != Cond_val2) {
                  System.out.println("value1 != value2");
                }
                if (Cond_val1 > Cond_val2) {
                  System.out.println("value1 > value2");
                }
                if (Cond_val1 < Cond_val2) {
                  System.out.println("value1 < value2");
                }
                if (Cond_val1 <= Cond_val2) {
                  System.out.println("value1 <= value2");
                }
```

```

                    in.nextLine();
                    System.out.println("Press Enter to go to Start");
                    in.nextLine();

                    break;

                  case 2:
                    System.out
                        .println("The && and || operators perform Conditional-AND and Conditional-OR "
                            + "operations on two boolean expressions. "
                            + "\nThese operators exhibit 'short-circuiting' behavior, "
                            + "which means that the second operand is evaluated only if needed.");

                    int value1 = 1;
                    int value2 = 2;
                    if ((value1 == 1) && (value2 == 2)) {
                      System.out.println("value1 is 1 AND value2 is 2");
                    }
                    if ((value1 == 1) || (value2 == 1)) {
                      System.out.println("value1 is 1 OR value2 is 1");
                    }

                    in.nextLine();
                    System.out.println("Press Enter to go to Start");
                    in.nextLine();

                    break;

                  default:
                    System.out.println("Invalid Choice\n back to main.");
                }

                in.nextLine();
                System.out.println("Press Enter to go to Start");
                in.nextLine();

                break;

              case 7:
                int array_menu;

                System.out.println("\nPlease select an array example: \n");
                System.out.println("1. find smallest value in array");
                System.out.println("2. find sum values of an array ");

                array_menu = in.nextInt();

                switch (array_menu) {
                  case 1:

                    int[] Minarray = new int[] { 10, 11, 88, 2, 12, 120 };
                    System.out.println("\nThe array values are: 10, 11, 88, 2, 12, 120");
                    // Calling MinVal() method for getting min value
                    int min = MinVal(Minarray);
                    System.out.println("Minimum Value is: " + min);
```

```

                        break;

                      case 2:
                        int[] sum_arr = new int[] { 21, 16, 86, 21, 3 };
                        System.out.println("\nThe array values are: 21,16,86,21,3");
                        int sums_fer_bums = sumVal(sum_arr);
                        System.out.println("The sum is " + sums_fer_bums);

                        break;

                      default:
                        System.out.println("Invalid Choice\n back to main.");
                    }

                    in.nextLine();
                    System.out.println("Press Enter to go to Start");
                    in.nextLine();

                    break;

                  case 8:

                    setUndget();

                    in.nextLine();
                    System.out.println("Press Enter to go to Start");
                    in.nextLine();

                    break;

                  case 9:

                    arrTwo();

                    in.nextLine();
                    System.out.println("Press Enter to go to Start");
                    in.nextLine();

                    break;

                  case 10:

                    in.nextLine();
                    System.out.println("Press Enter to go to Start");
                    in.nextLine();

                    break;

                  case 11:
                    System.out.println("\nBye-bye!");
                    quit = true;
                    break;

                  default:
                    System.out.println("Invalid choice.");

```

```
      }

    } catch (IndexOutOfBoundsException e) {
      System.out.println("Please input a Number as told by the menu\n");
    }

  } while (!quit);

  System.out.println("Hope to see you later!");
  // this closes the scanner

  in.close();

}

public static void setUndget() {
  Student tyler = new Student("Tyler"); // creating an instance of the Student class
  Student derek = new Student("Derek");

  tyler.setScore(1, 100);
  // creating an array capable of storing objects created from the Student class
  Student[] classroom = new Student[2];
  classroom[0] = tyler;
  classroom[1] = derek;

  for (Student aStudent : classroom) {
    // for each loop to go through array holding Student objects
    System.out.println(aStudent.getName());
    double[] studentScores = aStudent.getScores();
    for (double score : studentScores) {
      // for each loop to go through the array that is a field of the Student objects
      System.out.println(score);
    }
  }
}

public static void searchArr() {
  // this searches the array box for a value other than 0

}

public static void addmultOpperations() {
  System.out.println("Mathematical operations");
  System.out.println("java is capable of performing simple operation without"
      + "\n\tthe need for a different class to be called beforehand");
  System.out.println("for example simple arithmetic like addition, subtraction"
      + ", \n\tmultiplication and division with int, long, float, and double " + "data
types.");
  System.out.println(
      "\nFor example we will add the two integers 5 and 2 together to demonstrate
addition");
  int firstnum = 5;
  int secondnum = 2;
  firstnum += secondnum;
  System.out.println("The total of the equation is: " + firstnum);
  System.out.println(
      "\nwe will now multiply two double (.5 and 8.00) inputs together to demonstrate
```

```
multipolication");
    double thirdtnum = 8.00;
    double fourthdnum = 0.50;
    fourthdnum *= thirdtnum;
    System.out.println("The total of the equation is: " + fourthdnum);
  }

  public static void intDivision() {

    System.out.println("Integer division");
    System.out.println("java is capable of performing simple operation without"
        + "\n\tthe need for a different class to be called beforehand");
    System.out.println("for example simple arithmetic like addition, subtraction"
        + ", \n\tmultiplication and division with int, long, float, and double " + "data types.");
    System.out.println("\nThis example will demonstrate division of 10 by the ingeter 2: ");
    int divide = 10;
    divide /= 2;
    System.out.println("The total of the equation is: " + divide);

  }

  /**
   * This is primed to demonstrate the error handling from user inputs
   *
   *
   */
  public static void ifelseStatement(int thisIsAChoice) {

    int ifChoice = thisIsAChoice;
    if (ifChoice % 2 == 0) {
      System.out.println(" this number is even");
    } else {
      System.out.println(" this number is odd.");
    }

  }

  /**
   *
   * byte: Thebytedata type is an 8-bit signed two's complement integer. It has a minimum value of
   * -128 and a maximum value of 127 (inclusive). short: Theshortdata type is a 16-bit signed two's
   * complement integer. It has a minimum value of -32,768 and a maximum value of 32,767
   * (inclusive). int: By default, theintdata type is a 32-bit signed two's complement integer,
   * which has a minimum value of -231and a maximum value of 231-1. You can use the int data type to
   * represent an unsigned 32-bit integer, which has a minimum value of 0 and a maximum value of
   * 232-1. long: Thelongdata type is a 64-bit two's complement integer. The signed long has a
   * minimum value of -263and a maximum value of 263-1.You can use the long data type to represent
   * an unsigned 64-bit long, which has a value of 0 to 264-1. float: As with the recommendations
   * for byte and short, use afloat(instead of double) if you need to save memory in large
```

```java
arrays of
   * floating point numbers. This data type should never be used for precise values, such as
   * currency. double: For decimal values, this data type is generally the default choice. As
   * mentioned above, this data type should never be used for precise values, such as currency.
   * boolean: Thebooleandata type has only two possible values:trueandfalse. char: Thechardata type
   * is a single 16-bit Unicode character (i.e. letters and other symbols). It has a minimum value
   * of'\u0000'(or 0) and a maximum value of'\uffff'(or 65,535 inclusive).
   *
   */

  public static void dataType() {

    System.out.println("Java data Types:byte, short, int, long, float, double, boolean, char");
    System.out.println("\tFor a more detailed look at the data types"
        + "please look into the comments of the program");

  }

  /**
   *
   * @param Minimum
   *          this for loop counts all the array values and compares them to each other to get min
   * @return
   *
   */
  public static int MinVal(int[] Minimum) {
    int MinValue = Minimum[0];
    // this for loop counts all the array values and compares them to each other to get min
    for (int Min_count = 1; Min_count < Minimum.length; Min_count++) {
      if (Minimum[Min_count] < MinValue) {
        MinValue = Minimum[Min_count];
      }
    }
    return MinValue;
  }

  /**
   *
   * @param Summations
   *          this loop looks into the sum of all values in the array and adds them to the counter.
   * @return
   */
  public static int sumVal(int[] Summations) {
    int sum_total = 0;
    // this loop looks into the sum of all values in the array and adds them to the counter.
    for (int counter_arr = 0; counter_arr < Summations.length; counter_arr++) {
      sum_total += Summations[counter_arr];
    }
    return sum_total;
  }

  public static void arrTwo() {

    // declaring and initializing 2D array
    int arr[][] = { { 1, 2, 3 }, { 4, 5, 6 }, { 7, 8, 9 } };

    // printing 2D array in the form of a table.
    for (int i = 0; i < 3; i++) {
      for (int j = 0; j < 3; j++) {
        System.out.print(arr[i][j] + " ");
      }
      System.out.println();
    }
  }

}
```

```

public class Student {

  private String name;
  private double[] scores; // an array as the field of a class

  public Student(String n) { // constructor method, called automatic when new objects are created
    name = n;
    // actually create the space in memory for the array
    scores = new double[3];

  }

  public void setScore(int testNum, int score) {
    if (testNum >= 1 && testNum <= 3) {
      scores[testNum - 1] = score;
    }
  }

  public String getName() {
    return name;
  }

  public double[] getScores() {
    return scores;
  }
}
```