Introduction to Computer Programming

Lecture #9 – Programming Along the Straight and Narrow: Working With Arrays

Grouping Data

- We don’t buy individual eggs; we buy them in units of 12 (“dozens”).
  - We often think in terms of these groups and not the individual members.
  - Examples - classes, baseball teams, encyclopedia sets, etc.
- It is extremely helpful to be able to group data items that are closely related, e.g., class grades on a test, pay rates for a group of employees, etc.
Declaring Arrays

• Instead of writing:
  ```java
  int x;
  ```
  we can write:
  ```java
  int[] x = new int[10];
  ```
  the name \(x\) refers to the collection (or **array**) of integer values, which can contain up to 10 values.

Using An Array

• We can assign a value to any element in the array by specify the array by name and its index:
  ```java
  x[0] = 87;  // lowest index
  x[1] = 90;
  ...
  x[9] = 93;  // highest index
  ```
Using An Array (continued)

• An index can be any integer or character literal, constant, variable or expression:
  \[ x[five] = 34; \]
  \[ x[i+1] = x[i] + 3; \]
• This is really useful, because we do not want to have to write separate statement to assign values to each array element.

Using a Counting Loop To Set An Array

• Counting loops are really useful when manipulating arrays:
  \[ \text{for (i = 0; i < 10; i++)} \]
  \[ x[i] = \text{keyb.nextInt();} \]
import java.util.Scanner;

public class ArrayStuff {
    final static int numGrades = 10;

    // Find the class average on a test
    public static void main(String [] args) {
        // Our array
        int[] grades = new int [numGrades];
        int average;

        // Read the grades
        getGrades(grades);

        // Find the average
        average = calcAverage(grades);
        // Print the average and the grades
        printResults(grades, average);
    }

    //getGrades() - Read in the grades
    public static void getGrades(int[] grades) {
        Scanner keyb = new Scanner(System.in);
        int count;

        // Each time the loop, read another grade
        // into the array
        for (count = 0; count < numGrades;
             count++) {
            System.out.println("Enter a grade ");
            grades[count] = keyb.nextInt();
        }
    }
}

A Program To Find Class Average
// calcAverage() - Add up the grades, divide
//                  by the number of grades to
//                  find the average
public static int calcAverage(int[] grades) {
    int count, sum = 0;
    for (count = 0; count < numGrades; count++)
        sum = sum + grades[count];
    return(sum/numGrades);
}

// printResults() - Print the average and
//                  the grades
public static void printResults(int[] grades,
                                 int mean) {
    int i;
    System.out.println("The grades are:");
    for (i = 0; i < 10; i++)
        System.out.println(grades[i]);
    System.out.println("The average is " + mean);
    System.out.println("corresponding to a grade of "+ letterGrade(mean));
}
// letterGrades() - Translate the score into a letter grade
//
public static char letterGrade(int score) {
    if (score >= 90)
        return('A');
    if (score >= 80)
        return('B');
    if (score >= 70)
        return('C');
    if (score >= 60)
        return('D');
    return('F');
}

Example - A Payroll Program

• Let’s rewrite our payroll so it can use different tax brackets which the user will input.
The Revised Payroll Program

```java
import java.util.Scanner;

public class Payroll {
    // # of tax brackets
    final static int numBrackets = 5;

    // A payroll program that allows the user to
    // enter the tax brackets
    public static void main(String[] args) {
        double hours, rate;
        double gross, tax, net, trate;
        double[] taxmin = new double[numBrackets],
                taxrate = new double[numBrackets];
        boolean again;

        //Get the tax brackets before processing
        //any payroll records
        getBrackets(taxmin, taxrate);
        do {
            //Get the inputs, calculate the gross
            hours = getHours();
            rate = getRate();
            gross = getGross(hours, rate);

            //Calculate the tax and subtract it to get
            //the net
            tax = getTax(gross, taxmin, taxrate);
            net = gross - tax;
        } while (again);
    }
}
```
// Write the paycheck
writeCheck(gross, tax, net, rate, hours);
// Does the user want to process another
// record?
again = calcAgain();
} while (again);
// If not, quit

// getbrackets() - Input the tax brackets
// There are two different
// arrays - one stores the
// minimum gross
// for the tax bracket, the
// other stores the tax rate
public static void getBrackets
    (double[] taxmin, double[] taxrate) {
    Scanner keyb = new Scanner(System.in);
    int i;

    for (i = 0; i < numBrackets; i++) {
        System.out.println
            ("What is the maximum income for "+ " bracket ">#" + i+1 + "?");
"
taxmin[i] = keyb.nextDouble();
System.out.println
   ("What is the tax rate for bracket "
   + " #" + (i+1) + "?");
taxrate[i] = keyb.nextDouble();
taxrate[i] = taxrate[i]/100;
}
System.out.println("\n\n");

// getinput() - Input the hours worked and pay
// rate for each employee
public static double getHours() {
    Scanner keyb = new Scanner(System.in);
double hours;
    System.out.println("How many hours worked?");
hours = keyb.nextDouble();
return hours;
}

gpublic static double getRate() {
    Scanner keyb = new Scanner(System.in);
double rate;
    System.out.println("At what rate per hour?");
    rate = keyb.nextDouble();
    return rate;
}
//calcGross() - Calculate the gross pay
//              including any overtime
public static double getGross
    (double hours, double rate) {
    if (hours > 40) // With overtime
        return(40*rate + 1.5*rate*(hours-40));
    else // Without overtime
        return(rate*hours);
}

//getTax() - Calculate the tax for the
// employee using the tax brackets
public static double getTax(double gross,
    double[] taxmin, double[] taxrate) {
    int i;

    // If the employee doesn't make enough for the
    // lowest bracket the tax is zero
    if (gross < taxmin[0])
        return(0.0);
    // Find the appropriate bracket for the
    // employee
    for (i = 1;  i < numBrackets;  i++) {
        if (gross < taxmin[i])
            return(taxrate[i-1]*gross);
    }
//The employee is in the highest bracket
return(taxrate[numBrackets-1]*gross);
}

//writecheck() - Write the paycheck
public static void writeCheck(double gross,
   double tax, double net,
   double rate, double hours)  {
    // Print the input data
    System.out.printf("Hours = %4.2f\tRate = $%6.2f\n", hours, rate);
    // Print the gross pay
    System.out.printf("Gross = $%7.2f\n", gross);

    // Print the tax
    System.out.printf("Tax = $%7.2f\n", tax);

    // Print the net pay
    System.out.printf("Net = $%7.2f\n\n\n", net);
}
//calcAgain() - Returns true if the user want
// to go again.
// Returns false if not
public static boolean calcAgain() {
    Scanner keyb = new Scanner(System.in);
    String inputString = new String();
    char answer;

    do {
        System.out.println("Do you want to process another"
                + " payroll record (y - n)?");
        inputString = keyb.next();
        answer = inputString.charAt(0);
    } while (Character.toUpperCase(answer) != 'Y'
               && Character.toUpperCase(answer) != 'N');

    if (Character.toUpperCase(answer) == 'Y')
        return(true);
    else
        return(false);
}
Sorting

• What if the brackets are not read in the proper order?
• Let’s sort them and get them in the right order

Selection Sort

The algorithm:
1. Find the ith smallest number and place it in the ith slot
Selection Sort Algorithm Refined

The algorithm:

1. Find the ith smallest number and place it in the ith slot

1. For I = 0 to Size-1:
   1.1 Find the Ith smallest number
   1.2 Place it in slot I

Selection Sort

```java
import java.util.Scanner;

public class Sorting {
    final static int size = 5;

    public static void main(String[] args) {
        Scanner keyb = new Scanner(System.in);
        int i;
        int[] a = new int[size];

        for (i = 0; i < size; i++) {
            System.out.println("Enter a[" + i + "]\t?");
            a[i] = keyb.nextInt();
        }
    }
}
```
sort(a, size);
for (i = 0;  i < size;  i++)
    System.out.println("a[" + i + "] = "
        + a[i]);

// sort() - Sort an array of numbers
public static void sort(int[] x, int n)  {
    int i, j, small, index, temp;

    // Place the smallest number in the first
    // position
    // Place the second smallest in the second
    // position and so on.
    for (i = 0;  i < size -1;  i++) {
        small = Integer.MAX_VALUE;
        index = -1;
        // Compare each number that is not in its
        // proper place to the smallest so far
        for (j = i;  j < size;  j++)
            if (x[j] <small) {
                small = x[j];
                index = j;

    }
// Swap the ith smallest number into its proper place

temp = x[i];
x[i] = x[index];
x[index] = temp;