

Semester project

Final version due before pre-final quiz; versions 1 and 2 due earlier

You are to design, code in Java, and test a simple menu-driven item-management application, as specified below.

Your project will consist of documentation files (e.g., diagrams and text) and Java program files. Be sure to back up your work frequently to avoid loss of data in case of mishaps. Three ways to back up data are: to save copies of all work both on a laptop and on a network server (the Y: “drive” at FSC); to save on a flash memory stick; and to email files to yourself.

Project specification

1. Your application will allow the user to manage information about a collection of items (*records*, in database terminology; *object*, in object-oriented design terminology), stored in a disk file.

An item has attributes (fields); for example, a student item may have a student ID, a name, an email address, and a major. The following are some items you might consider choosing from:

- Sports team member;
- Product;
- Customer;
- Course;
- DVD.

For the item category you choose, select a set of attributes that can each be stored as a number or a string. One of the attributes should be a unique identifier such as a student ID or a course number.

2. The data will initially be stored in a disk file with one record per line. Create your data file using a text editor such as Notepad. The attribute values are each to be in double quotation marks, and commas separate the fields in a record. For example, if the file stores data about college courses, and the attributes of a course are the course number and description, then a file storing data on three courses might read as follows:

```
"63.152","Computer Science I"  
"63.135","Information Technology and Society"  
"63.460","Theory of Computing"
```

3. Your application will display a menu of user options. The initial version will manage just one item and will have the following choices:

- Retrieve item from disk;
- Display item;
- Update item;
- Save item to disk;

Your application will perform, or allow the user to perform, the operation corresponding to the chosen menu item. Display an error message if user requests output of data before inputting or retrieving data.

Documentation

Begin by writing *several clear paragraphs* in your own words that summarize:

- what the program does (specification), i.e., how the program interacts with the user;
- how it works internally (design).

Also include

- a flowchart of the main-menu loop described in the specification above;
- a module hierarchy for the application.
- UML class diagrams.

For each of the three versions described below, when you write the code, document your code to state your name, the file name, the date, what the purpose of the program is, and the purpose of each method.

Implementation (coding)

Write the code for this project in three versions, saving each version after testing it and saving test results. *Test results* must show that the code works correctly for all menu options and a variety of input data.

Initially your project will consist of separate programs, one to display the main menu and one for several operations. Later the separate programs will be integrated, each becoming a Java method – the menu method and the operations methods called from the menu method.

Version 1

1. Write a program that repeatedly displays a menu (see #3 under “Specification” above), retrieves a one-character menu choice, and displays a message echoing the menu choice. Save this program for later.

2. Write a program to read an item from disk, display it, update it from the keyboard, and save it to the disk.

The *Display* operation should display an item in a tabular format, as in the example below:

Last name	phone	email
Hopper	212-200-5432	hopper@grace.com

Version 2

Write a program that reads a *series* of data items from disk and displays them in a table, with headings as in Version 1.

Version 3

1. Separate the program for #2 of Version 1 into four Java methods, documenting each one. Copy the program for #1 into *main* and call the appropriate method in response to each choice of a menu option.

2. Under the *Update* option, code a sub-menu allowing the user to choose which attribute of the item to update.

3. Encode the data item attributes as a class. In the example above, the record class would have three data members.

Version 4

Extend your application to store a *collection* of items as an array of objects and to maintain a disk file with multiple items. In that version, the application will have the following menu choices:

- *New item* (append to file);
- *Search* by ID (if found, display the item and display a menu to allow update or delete the item);
- *Display all items* in the array collection, including summary information (turn your program for Version 2 into a method);
- Generate a report file in the same format as used by the display option.

Final submission

Submission should include each version, 1, 2, and 3 as above. Submit your documentation, source code, test results, and a sample data file.

The final submission is to be divided into sections on *specification*, *design*, *code*, and *testing*. Use dividers in a thin (not loose-leaf) binder.

Grading criteria for final submissions

Specification documentation	15%
Design documentation	15
Item data definition	15
Collection data definition	10
Control structures	15
Modules	15
Testing	15

Notes

All students are to work independently and submit their own code and documentation.

The project is designed to be done in stages. It is not acceptable to skip these stages and submit only the entire project at the end of the semester.

You may choose to add features to the above specification. Be sure to document these.

Submission dates

Documentation (six bullets listed under “Documentation”, over):

Wednesday, October 1

Code, Version 1: Wednesday, October 22

Code, Version 2: Thursday, November 6

Code, Version 3: Wednesday, November 26

Final submission (including code, Version 4):

Monday, December 8

Group work on topic 1 (Web)

Post all group-work solutions as replies to the problem description posting, with a subject line that specifies the problem solved and that lists names of group members who participated significantly in solving the problem.

For items A and C, each group will solve one problem (matching its group number).

- A. Write and test one HTML file (corresponding to your group number) with JavaScript with inputs and outputs are as specified below. Be sure to comment the file thoroughly, including the HTML file name and names of contributors.
- | <i>numeric input</i> | <i>output</i> |
|-----------------------------|---|
| 1. two numbers | their sum |
| 2. one number, x | x^3 |
| 3. two numbers, a and b | $a^2 + b$ |
| 4. two numbers | their product |
| 5. two numbers | their quotient |
| 6. two numbers | their difference |
| 7. threenumbers | the first two minus the third |
| 8. three numbers | the first multiplied by the sum of the others |
| 9. three numbers | the product of the first two, added to the third |
| 10. three numbers | the sum of the first two, multiplied by the third |
| 11. three numbers | the difference between the first two, multiplied by the third |
- B. See example file *count-yes.htm*. Modify it to create a web page that works as follows: Four buttons are displayed: “\$1”, “\$5”, “\$10”, “Done”. The user presses the 1, 5, and 10 buttons each a certain number of times, to request these amounts. After pressing “Done,” the user sees a number reflecting total amount requested. To solve this problem, you need to use an expression that assigns to *OnClick* a string that starts with 'alert(', ends with ')', and contains an expression that computes the sum of the \$1 amounts, \$5 amounts, and \$10 amounts.
- C. See the last section of the JavaScript handout and the example, *power.htm*, which uses the *while* statement in JavaScript. The following problems may be solved by modifying *power.htm*.
1. Prompt for an input x and output the sum of all numbers from 1 to x
 2. Prompt for an input and output its factorial.
 3. Prompt for an input and output the integer part of its base-2 logarithm, i.e., the number of times the number can be divided by two before reaching a value less than one.
 4. Input a and b , output a^b
 5. Input a series of numbers, output their sum
 6. Input a series of numbers, output largest
 7. Input a series of numbers, output smallest
 8. Input a series of numbers, tell whether they are all the same
 9. Input a series of numbers, tell whether they are in ascending order
 10. Input a series of numbers, tell whether any consecutive ones are duplicates.

Group work 2 (Program design)

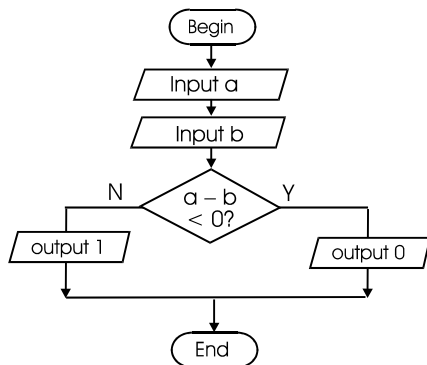
Each group is to solve the problem in each of the problem sets A .. E, corresponding to its group number*j*. Wrap around if the group number is larger than the highest problem number, e.g., under B, group 8 should do #1. Submit solutions as replies to the problem statement at Discussion Board Topic 2. Diagrams may be submitted electronically or on paper.

A. UML class diagrams

Write a UML class diagram, with name, attributes, and operations, for an object design or specification to support representation of

1. college students
2. college courses
3. film DVDs
4. songs
5. items of clothing for sale
6. job positions
7. corporations
8. stock offerings
9. group-work problems
10. professors

B. Tracing a flowchart (branch)



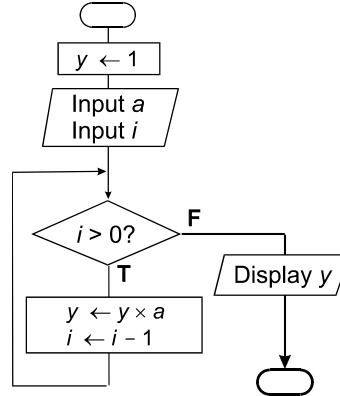
Consider the flowchart above. For each of the input pairs (a, b) , shown below, show the resulting output.

	Input a	Input b	Output
1.	2	1	_____
2.	1	3	_____
3.	4	2	_____
4.	7	10	_____
5.	3	2	_____
6.	3	5	_____
7.	6	2	_____

(Optional) Informally, what does the output of this algorithm tell about the two inputs? Answering this may require doing more than one problem.

C. Tracing a flowchart (loop)

Use the table below to trace the algorithm specified in the flowchart below for the given inputs of a and i .



1. 2 and 3;
2. 4 and 2;
3. 5 and 1;
4. 10 and 3;
5. 12 and 2;
6. 1 and 6;
7. 2 and 4.

a	i	y	output
_____	_____	_____	_____
_____	_____	_____	_____
_____	_____	_____	_____
_____	_____	_____	_____
_____	_____	_____	_____

D. Tracing pseudocode

Trace the following pseudocode, which counts spaces in a string, x . Note that x_i denotes the i th character of the string x and $|x|$ denotes the length of x .

```
input  $x$ 
 $y \leftarrow 0$ 
 $i \leftarrow 1$ 
while  $i \leq |x|$ 
    if  $x_i = ' '$ 
         $y \leftarrow y + 1$ 
     $i \leftarrow i + 1$ 
display  $y$ 
```

1. "hello there"
2. "this is me"
3. "1 * 2 * 3"
4. "no no no"
5. jane doe
6. john q. smith

E. Designing a solution

For the problem below matching your group number, design an algorithm or interactive process that does the following, using a flowchart or pseudocode.

Trace with sample data.

1. loops to accept four numbers and displays their sum
2. accepts two numbers and displays the larger.
3. accepts three numbers and displays the largest;
4. " with four numbers;
5. " with five numbers
6. prompts for two integers and displays their quotient, using only subtraction; show an error message if the divisor is 0.
7. accepts a number x and displays the sum of all the whole numbers from 1 to x ;
8. accepts a number x and displays the product of all the whole numbers from 1 to x
9. prompts for two integers and display their product, using only addition and subtraction operations in your calculations;
10. prompts for a string and tells whether it contains any doubled-up characters;
11. inputs n and finds the sum of n inputs;
12. inputs n and finds the largest of n inputs;
13. inputs n and finds the smallest of n inputs;
14. inputs n and finds the average of n inputs;
15. inputs n and tells whether n inputs are in ascending order;
16. inputs n and tells whether any consecutive duplicates exist in n inputs;
17. inputs n and tells whether all of n inputs are the same;
18. finds the range (largest minus smallest).

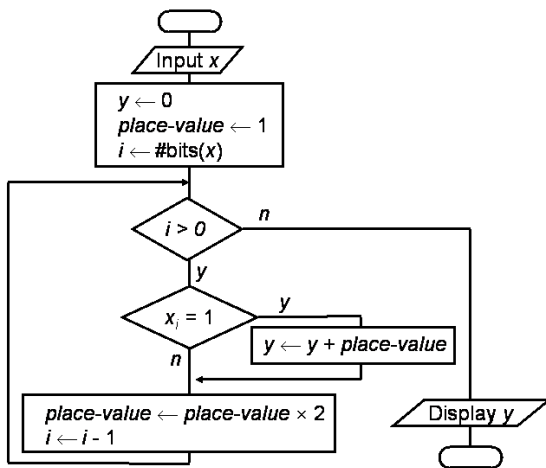
Group work 3 (Hardware)

Each group should solve the problem with the same number as the group. Post solutions, as replies to the problem spec message, with names of group members who contributed significantly.

A. Trace of binary to decimal conversion

Show a trace of this flowchart of binary-to-decimal conversion, for inputs

1. 11011;
2. 101101;
3. 101010;
4. 011001;
5. 010110;
6. 100101;
7. 100111



B. Binary to decimal conversion

Convert to decimal, showing your work:

1. 10110011
2. 10101110
3. 11011011
4. 10101110
5. 10110010
6. 01010011
7. 01100101
8. 10011101
9. 11001111

C. Assembler-program testing

Run the program *P10.java* on the assembler-language file, *xy.asm*. To do this, go into the directory with the P10 class files and type `java p10`. Enter the name of the assembler file, *xy*. Look at the file that traces the execution of *xy*. You will see assembler-language statements, alternating with the contents of registers and memory locations as detected after the execution of the assembler instructions.

- (a) From your observation, what occurs when the fifth line of the program, *store y*, executes?
- (b) Test the program for a few input values and post your test file, which will be 'xy.out'.
- (c) Based on your observation, write a formula that accurately describes the relationship between input and output of program *xy.asm*. (Your formula could be of the form "Output is *n* larger than input," or "Output is random," or "Output is same as input.")
- (d) Suggest a name for the program that describes what it does better than the name *xy.asm*, and better names for the data labels *x* and *y*.

D. Assembler programming (no loops)

For each expression below, write and test a program for ASM that inputs value *x* (or values *x*₁, *x*₂, etc.) and outputs the expression's value. *Hint*: for multiplication and division, use repeated addition or subtraction.

Edit your assembler code as a text file in *Notepad*, saving it with the extension ".asm". Test your code using the *P10* program used in part B.

Post solutions as attachments to replies to the problem message. Also post test results, which will be in files named the same as the program files, except ".out" replaces ".asm".

1. $x_1 + x_2 - (2x_3)$
2. $x_1 - 2x_2$
3. $2x_1 + x_2$
4. $3x_1 - 2x_2$
5. $2(x_1 + x_2)$
6. $2(x_1 - x_2)$
7. $x_1 + x_2 + x_3$
8. $x_1 + x_2 + x_3 - x_4$
9. $2(x_1 + x_2 + x_3)$

E. Assembler programming (loops)

Using the 10-instruction assembler language, solve one of the numbered versions below of the following problem:

Input x (or x_1 and x_2), and for any input, output the value of the numbered expression below.

Begin by writing a flowchart. *Suggestion:* See the example in the slides, 'count.asm', and work from that, noting how the 'jump', 'jump0' or 'jump-' instructions are used.

Expressions:

1. $20 * x_1$
2. $x_1 x_2$
3. x^2
4. $\lfloor x_1 \div x_2 \rfloor$ (integer quotient)
5. Remainder after computing quotient $x_1 \div x_2$
6. 0 if x is divisible by 2, 1 otherwise
7. The sum of all numbers from 1 to x
8. The largest of 5 inputs, looping repeatedly to use the name x for the input variable each time
9. The smallest of 5 inputs, using x as in #8.

Group work 4 (Java basics)

For all problems, comment your code with name of program file, assignment number, specification of program, names of students who contributed significantly to the solution. At end of program append test results.

A. Introductory Java assignment

Steps:

- Download the program `add.java` (Discussion Board, Topic 4, Examples), and compile and run it so that you are sure you know how to use the compiler (see handout, "Using Java Development Tools").
- Edit the program so that it inputs values to variables as indicated below and outputs the expression below that has the same number as your group. (Name the variables as you please.) Save the program under a new file name.
- Comment it so that it displays your group members' names at the top and so that it explains your code in your own words.
- Test the program with a couple of different sets of inputs, saving test results as a file. Text can be copied from the DOS window by expanding the command-line window (*Alt-Enter*) and pressing *PrtScr*.
- Post solutions as replies to this post or attachments. Also post test results (you may include test results as a long comment at the end of your program.)

Problems (expected output is shown below, given input of x or of x_1 and x_2):

- $x_1x_2 - 2x_3$
- $x_1 + 2x_2 - x_3$
- $2x_1 + x_2 + x_3$
- $3x_1 - 2x_2$
- $2(x_1 + x_2) + x_3$
- $x_1 + 2(x_2 - x_3)$
- $x_1 + x_2 + x_3$
- $x_1 + x_2 + x_3 - x_4$
- $2(x_1 + x_2 + x_3)$

B. Output of text

Write initials of group members in large letters vertically (see Horstmann, p. 178, Ex. P4.15).

Challenge: Write to a text file.

C. Algebra to Java conversions

Horstmann, p. 171, Ex. R4.1-4.3.

D. Computing values on two inputs (optional)

Horstmann, p. 174, Ex. P4.4 – no classes required.

E. Graphics (optional, may replace any of A to D)

Write a program to draw simple stick-figure pictures of the group members, labeled with their names. See Horstmann graphics chapter subsections for guidelines.

Group work 5 (Standard data types)

For all problems, comment your code with name of program file, assignment number, specification of program, names of students who contributed significantly to the solution. At end of program append test results.

A. Numeric types

Write a program to help you find the approximate largest numbers represented (without overflow) by one of the following types. One way to solve this problem would be to take an input value into a variable of the specified type, and see what the value output is. If the value output is wrong, then overflow has occurred.

1. *byte*
2. *short*
3. *int*
4. *long*

5. *float*
6. *double*

You may need to use several statements or several runs of your program to help home in on the highest value. See the textbook for guidance on numeric types and overflow.

B. Characters and strings

Input three strings: the user's first, middle, and last name. Store the first name, the middle initial and the last name as one string variable, including the necessary spaces and period, capitalizing each part of the name even if the user has not done so. Display the resulting string variable and state its length.

C. File I/O

Modify the program your group wrote for Group Work 4, problem set A, so that it will read program input from a text file and write output to a file.

Group work 6 (Control structures)

Document and test. Post solutions on the Discussion Board as replies to the problem postings. List names of students who contributed significantly to the solution.

A. Syntax errors

Horstmann, p. 217, Ex. R5.1. Each group do one problem, a=1, b=2, etc.

B. Writing loops in Java

Horstmann, pp. 221-223, Ex. P5.1-P5.10.

C. Converting flowcharts to Java code

See topic 2 (Design), group work problems D. Convert to Java the flowcharts you wrote for your group's problems.

D. Numeric conversions between bases

Implement conversion algorithms in Java of the kind below that corresponds to your group number:

1. Binary to decimal
2. Decimal to binary
3. Hexadecimal to binary
4. Binary to hexadecimal
5. Hexadecimal to decimal
6. Decimal to hexadecimal

E. String manipulation

1. Search an input string for an input character, display the character in brackets within the string if it is found, otherwise display "not found in " and the string.
2. Find the longest run of the same character in input string, display it in brackets.
3. Find the last period in an input string, bracket it in output.
4. Input a name in two or more strings, display "Hello " and the first name only.
5. Input a string and display a count of the number of vowels found.
6. Input a string and upper-case its characters for output.
7. Check an input string and tell whether it is a palindrome (spells same way backwards as forwards).

F. File reading

Code in Java the program design your group wrote for Group Work 2, problem set D. Write it to read program input from a text file. (For sample file-reading code, see handout and Java file *read_file.java*.)

Group work 7 (Methods and classes)

Post well documented solutions on the Discussion Board as replies to the threads for the problems. List names of students who contributed significantly to the solution.

A. Defining simple method

Write a program that calls from *main* a method that displays the attributes of some object; for example, a student's name and favorite sport. The attributes should be variables declared in *main*.

Comments should include specification of the problem and names of students who participated. At the end of the Java code, paste in some test results as a comment. Post solution as a reply to this post.

B. Methods sharing member data

See specs for problem A. In your application class, declare private members to store the chosen attributes. In *main*, call a method that inputs the attributes, then call a method that displays them. Define the methods as class members.

C. Parameter passing

Write a program that accepts two numeric inputs, x_1 and x_2 , and calls a method that accepts them as parameters and displays their

1. Minimum
2. Average
3. Maximum
4. Distance (absolute value of difference)
5. Real quotient x_1 / x_2 .
6. Remainder after dividing x_1 by x_2 .

D. Return values

See group work C. Modify to have *main* display the value calculated and to have the new method you wrote pass back the value calculated as a return value.

E. Fix 'extract_word.java'

See the program below. It has errors but gives output that helps point the way to fixing its bug(s). Find the bugs and fix them so that the program works correctly. Use your own test data for file "extract_word.txt".

```
/* extract_word.java:
   Reads a string from file 'extract_word.txt',
   prompts for number, n, displays the nth word
   of the string.   D. Keil 4/08 HAS BUGS */
import java.util.Scanner;
import java.io.FileReader;

public class extract_word
{
    private static String nth_word(String s, int n)
    // Returns nth word of s as delimited by spaces.
    {
        // Find nth space:
        int space = 0, j = 0;
        for (int i=0; i < n; i++)
        {
            // Find location of next space:
            for (j = j+1; s.charAt(j) != ' '; j++)
                ;
            System.out.println("space found at " + j);
        }
        int word_starts = j+1;
        // Find length of word:
        int len = 0;
        for ( ; s.charAt(word_starts+len) != ' '; len++)
            ;
        System.out.println("word starts at " +
            word_starts + " and length= " + len);
        return s.substring(word_starts, word_starts+len);
    }

    public static void main(String[] args)
    throws FileNotFoundException
    {
        // Open input file, read a string and display:
        System.out.print("Reading 'extract_word.txt'");
        FileReader reader = new FileReader
            ("extract_word.txt");
        Scanner file_in = new Scanner(reader);
        String buf = file_in.nextLine();
        file_in.close();
        System.out.println("File contents:\n"+buf);

        // Prompt for number:
        System.out.print("Enter # of word: ");
        Scanner keyboard = new Scanner(System.in);
        int n = keyboard.nextInt();

        // Display nth word:
        String word = nth_word(buf, n);
        System.out.println("The #" + n +
            " word of \"" + buf + "\" is " + word);
    }
}
```

F. Defining a Java class

Define a Java class to implement the multiple-attribute entity you used for problem A. For example, if your class were a student sports fan, the class would have at least two attributes: student name and name of favorite sport. Write a constructor and methods to input and output instances of your class.

Group work 8 (Arrays)

1. See handout, “A collection of points,” which gives C++ code for a collection of points on an (x,y) -coordinate grid, using an array of objects. It retrieves from a text file a set of points, coded as pairs of integers.

Write a Java program that does the same thing using a point class and a second class that implements a collection of points.

2. See handout, “An array of objects: a basketball league,” which gives C++ code for a collection of basketball teams with info on most recent game scores.

Write a Java program that does the same thing using a team class and a second class that implements a collection of points.

Group work 9 (File maintenance)

A. Entity-relationship design

Referring to the slide on Entity-Relationship Design, design a database to represent three entities, each entity represented by one table with a primary key. The third entity represents a joining of one instance of each of the other two entities in a relationship, such as customer and product purchased by customer. Design three-table databases for the following:

1. job applicants being hired for jobs, so that a sheet can be generated with information about all the new hires for a given period of time;
2. poems being published in journal issues, so that a sheet can be generated listing all the poems published for a given journal issue;
3. attendees signing up for workshops at a research conference, so that a report can be generated listing all the attendees at a given workshop;
4. home buyers visiting open houses, so that a list can be generated of all the home buyers who visited a given house.
5. customers buying CDs at music stores, so that a sales slip can be generated;
6. CD vendors selling shipments of music to music stores, so that a vendor can list all the shipments made to a certain music store.

B. Social and professional issues

Discuss in one or two paragraphs one of the following issues.

1. How are movies on the web different from journals on the web with respect to intellectual property?
2. Should plagiarism be criminalized? (Defend or refute.)
3. What is ethics?
4. Is there a special kind of ethics for the information age?
5. Should unauthorized software copying be criminalized? (Defend or refute.)
6. What is copyright and how does its application differ between electronic and other media?
7. Does cyberspace have unique features that justify different speech protections for sexual material?
8. Does the opportunity to broadcast negative messages to a large Internet audience preclude strong legal protection from defamation?
9. Is there a natural human right to privacy? (Defend or refute.)
10. What new privacy issues are raised by the information revolution?
11. What are the main technical and integrity concerns raised in the controversy over electronic voting?