

SOFTWARE DEVELOPMENT EXAMPLE

PROBLEM STATEMENT:

A college offers a course that prepares students for the state licensing exam for real estate brokers. Last year, ten of the students who completed this course took the exam. The college wants to know how well its students did on the exam. You have been asked to write a program to summarize the results. You have been given a list of these 10 students. Next to each name is written a 1 if the student passed the exam or a 2 if the student failed.

Your program should analyze the results of the exam as follows:

- 1. Input each test result (i.e., a 1 or a 2). Display the message "Enter result" on the screen each time the program requests another test result.*
- 2. Count the number of test results of each type.*
- 3. Display a summary of the test results indicating the number of students who passed and the number who failed.*
- 4. If more than eight students passed the exam, print the message "Raise tuition."*

DESIGN:

After reading the problem statement carefully, we make the following observations:

1. The program must process test results for 10 students. A counter-controlled loop can be used because the number of test results is known in advance.
2. Each test result has a numeric value either a 1 or a 2. Each time the program reads a test result, the program must determine whether the number is a 1 or a 2. We test for a 1 in our algorithm. If the number is not a 1, we assume that it is a 2
3. Two counters are used to keep track of the exam results one to count the number of students who passed the exam and one to count the number of students who failed the exam.
4. After the program has processed all the results, it must decide whether more than eight students passed the exam.

FIRST REFINEMENT

Let us proceed with top-down, stepwise refinement. We begin with a pseudocode representation of the top:

Analyze exam results and decide whether tuition should be raised

Once again, the top is a complete representation of the program, but several refinements are likely to be needed before the pseudocode can evolve naturally into a C# program.

Initialize variables

Input the 10 exam results, and count passes and failures

Print a summary of the exam results and decide whether tuition should be raised

Here, too, even though we have a complete representation of the entire program, further refinement is necessary. We now commit to specific variables. Counters are needed to record the passes and failures, a counter will be used to control the looping process and a variable is needed to store the user input. The variable in which the user input will be stored is not initialized at the start of the algorithm, because its value is read from the user during each iteration of the loop.

SECOND REFINEMENT

Initialize variables

Initialize passes to zero

Initialize failures to zero

Initialize student counter to one

Input the 10 exam results, and count passes and failures

While student counter is less than or equal to 10

Prompt the user to enter the next exam result

Input the next exam result

If the student passed

Add one to passes

Else

Add one to failures

Add one to student counter

Print a summary of the exam results and decide whether tuition should be raised

Print the number of passes

Print the number of failures

If more than eight students passed

Print "Raise tuition"

THIRD REFINEMENT

The complete second refinement of the pseudocode appears as:

```
1  Initialize passes to zero
2  Initialize failures to zero
3  Initialize student counter to one
4
5  While student counter is less than or equal to 10
6      Prompt the user to enter the next exam result
7      Input the next exam result
8
9      If the student passed
10         Add one to passes
11     Else
12         Add one to failures
13
14     Add one to student counter
15
16 Print the number of passes
17 Print the number of failures
18
19 If more than eight students passed
20     Print "Raise tuition"
```

The C# class that implements the pseudocode algorithm is:

```
public class Analysis
{
    public void ProcessExamResults()
    {
        // initialize variables in declarations
        int passes = 0; // number of passes
        int failures = 0; // number of failures
        int studentCounter = 1; // student counter
        int result; // one exam result from user

        // process 10 students using counter-controlled repetition
        while ( studentCounter <= 10 )
        {
            // prompt user for input and obtain a value from the user
            Console.Write( "Enter result (1 = pass, 2 = fail): " );
            result = Convert.ToInt32( Console.ReadLine() );

            // if...else nested in while
            if ( result == 1 ) // if result 1,
                passes = passes + 1; // increment passes
            else // else result is not 1, so
                failures = failures + 1; // increment failures

            // increment studentCounter so loop eventually terminates
            studentCounter = studentCounter + 1;
        } // end while

        // termination phase; prepare and display results
        Console.WriteLine( "Passed: {0}\nFailed: {1}", passes, failures );

        // determine whether more than 8 students passed
        if ( passes > 8 )
            Console.WriteLine( "Raise Tuition" );
    } // end method ProcessExamResults
} // end class Analysis
```