Compilers [Spring 2020]

Programming Assignment I

Objectives

1. To understand the definitions of DJ and DISM, which will serve as source and target languages for a compiler built in future assignments.
2. To implement small DJ and DISM programs.
3. To become familiar with and able to use a DISM simulator.

Due Date: Sunday, January 26, 2020 (at 11:59pm).

Machine Details

Complete this assignment by yourself on the cselx##.csee.usf.edu computers. You are responsible for ensuring that your programs compile and execute properly on these machines.

Assignment Description

For this assignment, you will acquaint yourself with the DJ and DISM languages by implementing one small program in each language. You will write a DJ program in a file called *binProd.dj* and a DISM program in a file called *binProd.dism*.

The desired functionality is for your programs to input pairs of natural numbers as they're entered by the user, until the user enters a zero as the first in a pair of numbers. The first number in each pair indicates a bin for the second number. For example, the pair of numbers 3 4 means that bin 3 contains the number 4. After the user enters a bin number of 0, your programs then allow the user to enter natural numbers until another 0 is entered; for every natural number *i* entered, your programs must output the product of all the numbers in bin *i* (or 0 if there are no numbers in that bin). Throughout all these operations, your programs must be reasonably efficient—there should never be long, noticeable pauses during execution.

Examples of Desired Behavior:

```
Enter a natural number: 1
Enter a natural number: 2
Enter a natural number: 1
Enter a natural number: 3
Enter a natural number: 1
Enter a natural number: 4
Enter a natural number: 0
Enter a natural number: 1
24
Enter a natural number: 0
```

(Here bin 1 contains 2, 3, and 4, so the product for bin 1 is 24)

```
Enter a natural number: 0
Enter a natural number: 2
0
Enter a natural number: 0
```

(Here no bins contain numbers, so 0 would be returned for all products)

```
Enter a natural number: 0
Enter a natural number: 0
```
(Here no numbers are stored and no products are sought)

```
Enter a natural number: 200000
Enter a natural number: 5
Enter a natural number: 200000
Enter a natural number: 0
Enter a natural number: 0
Enter a natural number: 200000
0
Enter a natural number: 0
```
(Here bin 200000 contains a 0, so 0 is returned for its product)

Hints

- Whenever a DJ or DISM program attempts to read a natural number, the prompts of "Enter a natural number: " get printed automatically. Hence, you don't need to worry about outputting those prompts. DJ and DISM programs can only input and output natural numbers (using the *readNat* and *printNat* calls in DJ, and the *rdn* and *ptn* instructions in DISM).
- You may assume that the user never stores more 1,000 numbers total into bins.
- My *binProd.dj* is 25 lines of code (not counting whitespace/comments), and my *binProd.dism* is 28 lines of code.

Testing Your DISM Program: Please use the DISM simulator, sim-dism, to test your DISM program. When your DISM program halts, it may halt with any code.

Testing Your DJ Program: Because you're writing a program in a new language, for which no compiler yet exists, you can't test your program by executing it! This situation is unpleasant but realistic. You'll have to ensure by hand that your DJ program is valid and would behave correctly if executed. You could, however, modify it into a valid Java program (using Java's Scanner class to mimic readNat), and then test that Java program.

Formatting, Grading, and Submission Notes

- To make it easier for our TA to read and grade code files, **use spaces rather than tabs** in your code and **avoid long lines of code** (I try to limit lines to 80 characters).
- Your programs will be graded on both correctness and style, so include good comments, well-chosen variable names, etc. For full credit, your code must not be significantly more complicated than necessary.
- The TA will test submissions on inputs not shown in the example above.
- Type the following pledge as an initial comment in every file you submit for this course: "I pledge my Honor that I have not cheated, and will not cheat, on this assignment." Type your name after the pledge. Not including this pledge in a submitted file will lower your assignment grade 50%.
- Upload and submit both of your files for Assignment 1 in Canvas.
- You may submit your assignment in Canvas as many times as you like; we will grade your latest submission.
- For every day that your assignment is late (up to 2 days), your grade reduces 10%.