CSCI 3155: Homework Assignment 8

Due Thursday, November 19, 2009

For this assignment, you will again work with a partner. You will write up and turn in this assignment in pairs. Choose one of you to upload your write up to the moodle. Please name the file

```
hw8-YourIdentiKey-YourPartnersIdentiKey.pdf
```

Also, please include both of your PL-Detective user ids on your write-up and indicate which one should be considered the submission id. As always, you are welcome to discuss in larger groups. Just be sure to acknowledge those with which you discussed.

Bookkeeping

Exercise 1: Indicate in a sentence or two how much time you spent on this homework, how difficult you found it subjectively, and what you found to be the hardest part. Any non-empty answer will receive full credit.

If you would like share something about yourself that I do not already know, please do so. And if your opinions have changed since the last submission, indicate one thing you like about the class so far and one thing you would change about it.

Arrays

Exercise 2: Skill 15.1. Assume that MYSTERY uses the following rule for array subtyping:

```
ARRAY I OF T <: ARRAY I OF U if T <: U
```

Give a MYSTERY program that demonstrates that this subtyping rule is invalid. Note: you may not be able to run your program in MYSTERY since in reality MYSTERY does not support this invalid rule.

Exercise 3: Synthesis. Languages differ greatly in how they handle arrays. In this question, we will focus on one issue with arrays: what happens when one array is assigned to another (assuming the two arrays are of compatible types). Consider, for example, two variables declared as follows in MYSTERY syntax:

```
VAR a : ARRAY [1 TO 5] OF INTEGER;
VAR b : ARRAY [1 TO 5] OF INTEGER;
BEGIN
```
What happens when we have an assignment \( a := b \)? If the assignment is “by reference” (which is how Java works), variables of array type are represented as pointers and the outcome of the assignment is that \( a \) ends up pointing to the same array that \( b \) points to (i.e., \( a \) and \( b \) are aliases for the same array). If the assignment is “by value” (which is how Modula-3 works), then the assignment copies the contents of array \( b \) to array \( a \).

1. Use this link to figure out if MYSTERY uses array assignment by reference or by value.

   http://www-plan.cs.colorado.edu/diwan/pldarray.htm

Your answer to this question should describe which one MYSTERY uses and also provide the evidence that enabled you to come to your conclusion. You submissions may execute up to a total of 6 PRINT statements. Additional attempts will be charged at 5% of this question's points per executed PRINT.

2. Discuss the relative advantages and disadvantages of array assignment by value and array assignment by reference.

Parameter Passing

Exercise 4: In this question you will discover the parameter passing mechanism for MYSTERY using this web page to access the PL-Detective.

http://www-plan.cs.colorado.edu/diwan/pldparam1.htm

For the purpose of this question assume that the actual’s type and the corresponding formal’s type must be the same using structural type equality. Also assume that MYSTERY uses static scoping.

1. Skill 16.1. For the following parameter passing implementations:
   - pass-by-value
   - pass-by-value-result
   - pass-by-reference
   - pass-by-name

give a realistic example for which the implementation is most suitable. You may want to consult Section 9.5.2 of the text. Support your answer with clear arguments.

2. Skill 16.2. Determine and describe the parameter passing mechanism in MYSTERY using the above link. Be sure to state how MYSTERY’s implementation addresses the issues discussed in part (i). Your submissions may execute up to 10 PRINTs. Each additional PRINT will be charged 5% of the points for the question.

3. Provide and discuss the evidence that supports your case.
Exercise 5:  

Exercise 6:  

Assume that you have a procedure with a single formal parameter of type $T$. Now, let’s suppose you want to call it with an actual parameter of type $U$. How should types $T$ and $U$ be related in order for the call to be safe with respect to types given each of the following parameter passing implementations. Clearly explain your answer.

1. Pass-by-value
2. Pass-by-result
3. Pass-by-value-result
4. Pass-by-reference
5. Pass-by-readonly-reference (i.e., same as pass-by-reference, but the argument cannot be modified, only read).
6. Pass-by-name