Rose-Hulman Institute of Technology
Computer Science and Software Engineering
Profs. Laurence Merkle and Archana Chidanandan

Homework 1 (corrected) (Assembly Language)

Directions

This assignment is due Thursday, September 18, 2003 for Sections 1 and 3 and Friday, September 19, 2003 for Section 2. Submit your solutions on a separate sheet of paper. *Hint*: Use SPIM.

Learning Objectives

In the process of completing this homework assignment, students will develop their abilities to

- Predict the actual assembly language instructions corresponding to a pseudoinstruction.
- Implement algorithms involving arrays, selection, iteration, and procedural abstraction in assembly language.

Problems

1. [10 pts] Hennessy and Patterson Problem 3.10.

2. [10 pts] The selection sort algorithm begins by finding the largest element of an array and exchanging it with the last element. It then finds the second largest element and exchanges it with the next to last element. It repeats this process until the array is sorted. The following Java code implements the algorithm:

```
public static void SelectionSort( int list[], int n ){

     for( int i = n; i > 1; i-- ){
          int m = list[ 0 ];
          int k = 0;

          for( int j = 1; j < i; j++ ){
               if( list[ j ] > m ){
                    m = list[ j ];
                    k = j;
               }
          }

          int temp = list[ i - 1 ];
          list[ i - 1 ] = list[ k ];
          list[ k ] = temp;
     }
}
```

Rose-Hulman Institute of Technology
Computer Science and Software Engineering
Profs. Laurence Merkle and Archana Chidanandan

Using P03-2.asm as a starting point, write a MIPS program that implements the selection sort algorithm. Be sure to document your program properly. Turn in a hard copy listing of your program.

3. [10 pts] Write a procedure *find* in MIPS assembly language, adhering to MIPS register usage conventions. The procedure should take ~~two~~ **three** arguments. The first argument is a pointer to an array of integers ~~and the second argument is an integer value~~, **the second is the number of elements in the array, and the third is an integer value**. *find* must find and return for the ~~second~~ **third** argument, the index value of its first occurrence in the integer array. If the integer value does not exist in the array, then *find* must return -1.