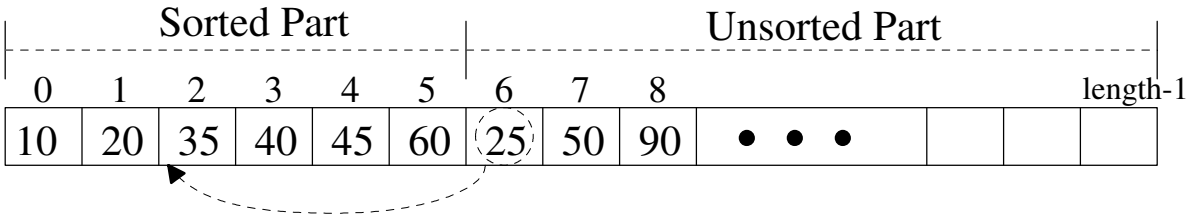
Computer Architecture HW #2

Due: Friday, Feb. 3 (3 PM in ITT 305 mailbox or under my office door, ITT 313)

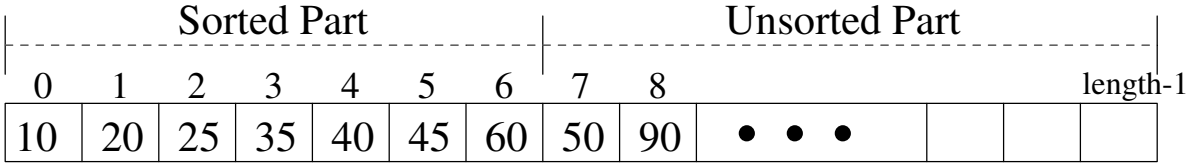
1. Another simple sort is called insertion sort. Recall that in a simple sort:

- the outer loop keeps track of the dividing line between the sorted and unsorted part with the sorted part growing by one in size each iteration of the outer loop. (below the firstUnsortedIndex keeps track of the dividing line)
  - the inner loop's job is to do the work to extend the sorted part's size by one.

After several iterations of insertion sort's outer loop, an array might look like:



In insertion sort the inner-loop takes the "first unsorted item" (25 at index 6 in the above example) and "inserts" it into the sorted part of the array "at the correct spot." After 25 is inserted into the sorted part, the array would look like:



Consider the following insertion sort algorithm that sorts an array numbers:

```
InsertionSort(numbers - address to integer array, length - integer)
  integer firstUnsortedIndex, testIndex, elementToInsert;

    for firstUnsortedIndex = 1 to (length-1) do

        testIndex = firstUnsortedIndex-1;
        elementToInsert = numbers[firstUnsortedIndex];

        while (testIndex >=0) AND (numbers[testIndex] > elementToInsert ) do
            numbers[ testIndex + 1 ] = numbers[ testIndex ];
            testIndex =  testIndex - 1;
        end while

        numbers[ testIndex + 1 ] = elementToInsert;

    end for
end InsertionSort
```

a) Where in the code would unconditional branches be used and where would conditional branches be used?

b) If the compiler could predict by opcode for the conditional branches (i.e., select whether to use machine language statements like: "BRANCH_LE_PREDICT_NOT_TAKEN" or "BRANCH_LE_PREDICT_TAKEN"), then which conditional branches would be "PREDICT_NOT_TAKEN" and which would be "PREDICT_TAKEN"?

c) Assumptions:

- length = 100 and the numbers are initially in **descending** order before the insertion sort algorithm is called
- the five-stage pipeline discussed in class
- the outcome of conditional branches is known at the end of the E stage
- target addresses of all branches is known at the end of the D stage
- ignore any data hazards

Under the above assumptions, answer the following questions:

i) If fixed predict-never-taken is used by the hardware, then what will be the total branch penalty (# cycles wasted) for the algorithm? (Here assume NO branch-prediction buffer)

ii) If a branch target buffer with one history bit per entry is used, then what will be the total branch penalty (# cycles wasted) for the algorithm? (Assume predict-not taken is used if there is no match in the branch-prediction buffer) Explain your answer.

iii) If a branch target buffer with two history bit per entry is used, then what will be the total branch penalty (# cycles wasted) for the algorithm? (Assume predict-not taken is used if there is no match in the branch-prediction buffer) Explain your answer.