Computation Structures — Assembly (part 2)

18 November 2020

Translate the following C functions in β-assembly.

1. Searching the maximum of an array:

```
int array_max(int* array, int size) {
  int pos = -1, max = 0;
  for (int i = 0; i < size; ++i) {
    if (pos == -1 || max < array[i]) {
      max = array[i];
      pos = i;
    }
  }
  return pos;
}
```

2. Partitioning an array:

```
int swap(int* a, int *b) {
  int tmp = *a;
  *a = *b;
  *b = tmp;
}

int partition(int* array, int size, int pivot) {
  // Swap pivot with the end of the array
  swap(array + pivot, array + size - 1);
  int pivot_val = array[size - 1];

  // Partition around pivot_val
  long small = -1;
  for (int curr = 0; curr < size - 1; ++curr) {
    if (array[curr] <= pivot_val) {
      small++;
      swap(array + curr, array + small);
    }
  }

  // Place pivot between two subarrays
  swap(array + small + 1, array + size - 1);
  return small + 1;
}
```

3. Sorting an array by selection sort:

```c
void selection_sort(int* array, int size) {
  for (int i = size - 1; i > 0; --i) {
    int max_index = array_max(array, i + 1);
    int tmp = array[i];
    array[i] = array[max_index];
    array[max_index] = tmp;
  }
}
```