Mahros Mohamed Mahros 223106831

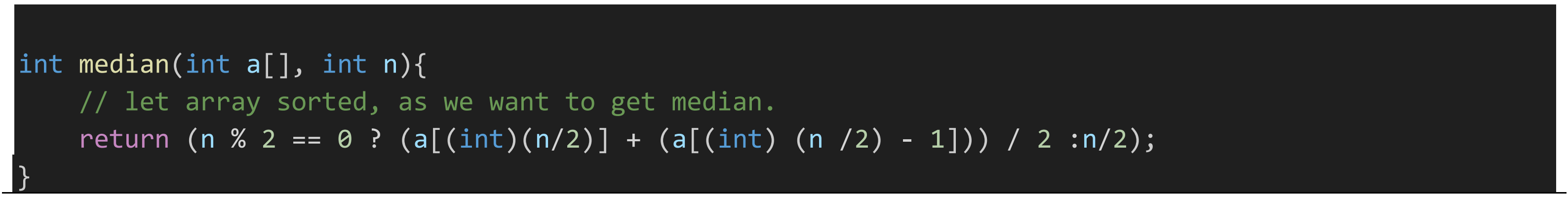
1. **Suppose the insertNextItem function is called with a negative value of i ¾ does the loop terminate? Explain. Can we be certain that i is always positive?**

- Yes, this loop will terminate as there is a condition that x >= 0;

2. **Trace the operation of insertionSort for a = {17, 3, 9, 6, 14, 25, 2}.**

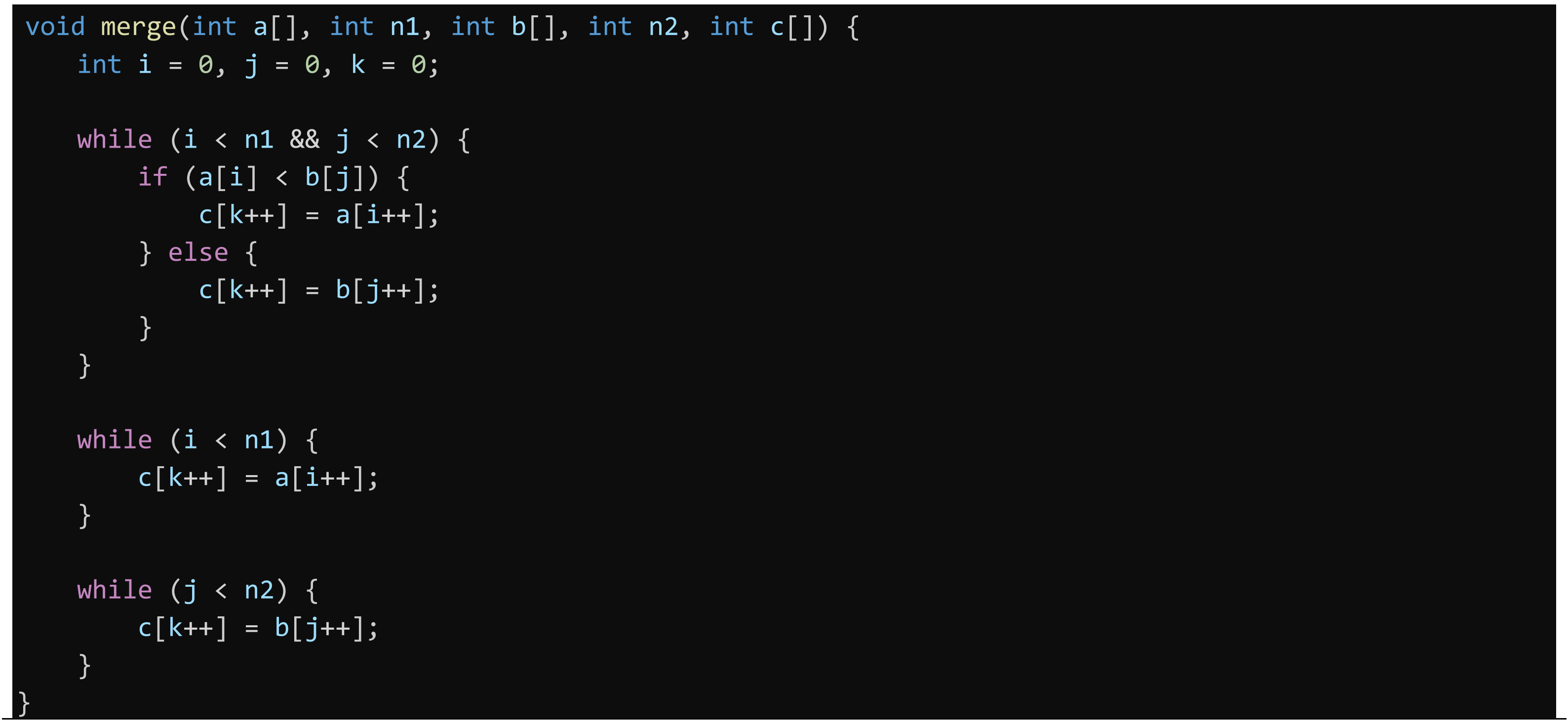
0	17	3	9	6	14	25	2
1	17	3	9	6	14	25	2
2	3	17	9	6	14	25	2
3	3	9	17	6	14	25	2
4	3	6	9	14	17	25	2
5	3	6	9	14	17	25	2
6	2	3	6	7	14	17	25

3. **Write a function median with the following specification:**



```
int median(int a[], int n){
    // let array sorted, as we want to get median.
    return (n % 2 == 0 ? (a[(int)(n/2)] + (a[(int) (n /2) - 1])) / 2 :n/2);
}
```

4. Write a function merge with the following specification:



```
void merge(int a[], int n1, int b[], int n2, int c[]) {
    int i = 0, j = 0, k = 0;

    while (i < n1 && j < n2) {
        if (a[i] < b[j]) {
            c[k++] = a[i++];
        } else {
            c[k++] = b[j++];
        }
    }

    while (i < n1) {
        c[k++] = a[i++];
    }

    while (j < n2) {
        c[k++] = b[j++];
    }
}
```

5. Write the following function:

```
void swapElements(int a[], int maxPos, int last){
    int temp = a[maxPos];
    a[maxPos] = a[last];
    a[last] = temp;
}
```

6. Rewrite the Selection Sort Algorithm using the minSelect function in place of maxSelect. In this algorithm, you will start by selecting the minimum number and put it in the beginning of the array, and then repeat the process.

```
int minIndex(int a[], int l, int r) {
    int mi = l;
    for (int i = l + 1; i <= r; i++)
        if (a[i] < a[mi])
            mi = i;
    return mi;
}

void selectionSort(int a[], int n) {
    int i, j, mi;

    for (i = 0; i < n - 1; i++) {
        mi = minIndex(a, i, n - 1);
        swap(a[mi], a[i]);
    }
}
```

7. Modify BubbleSortPhase and BubbleSort so that the sort terminates as soon as it knows that the array is sorted rather than completing all n-1 passes.

```
bool bubbleSortPhase(int a[], int n) {
    bool swapped = false;
    for (int i = 0; i < n - 1; i++) {
        if (a[i] > a[i + 1]) {
            swap(a[i], a[i + 1]);
            swapped = true;
        }
    }
    return swapped;
}

void bubbleSort(int a[], int n) {
    bool swapped;
    do {
        swapped = bubbleSortPhase(a, n);
        n--;
    } while (swapped);
}
```