Exercise 7-2: maxmin.c

Maximilian Fernaldy - C2TB1702

Note: some links and other HTML-related objects may not work in pdf form. Consider reading the webpage format of the report here.

In exercise 7-2, we are tasked to find the minimum and maximum values of an array. However, rather than creating two separate functions that each return a single value, we are to use a single function with access to variables declared in the main function. This can be done by using pointers, as they transcend scopes.

```
int main() {
    int max,min;
    int array[LEN];
    int i;
    // assign array
    for(i=0; i < LEN; i++){
        printf("array[%d] = ",i);
        scanf("%d", &array[i]);
    }
    /* find max-min value from array */
    maxmin_array(array, &max, &min);
    printf("max:%d min:%d\n", max, min);
    return 0;
}
```

In the main function, we initialize the variables `max` and `min` to contain the value of the maximum integers. Then we create and fill the array, call the `maxmin_array()` function and pass the array, along with the memory addresses of `max` and `min` so that the function can directly modify their values.

As for the function itself,

```
void maxmin_array(int array[], int *pmax, int *pmin) {
    // This function modifies max and min by using the pointers pmax and pmin.
    // For the array, we are essentially passing the pointer to the first element
    // by passing int array[] as an argument. Note that this can essentially be
    // replaced by passing int *array, but it's less readable in this context.

    // First set max and min to the first element
    *pmax = array[0]; *pmin = array[0];

    // The for loop can start from i=1 since initial value is index 0
    for (int i = 1; i < LEN; i++) {
        if (array[i] > *pmax) {
            *pmax = array[i];  // array[i] is the same as *(array+i).
        } else if (array[i] < *pmin) {
            *pmin = array[i];
        }
    }
}
```

In the function, we first initialize max and min using the pointers to them, by dereferencing the pointers:

```
*pmax = array[0]; *pmin = array[0];
```

To find the maximum and minimum values in the array, we iterate through it and check if the current number is either smaller than `min` or larger than `max`. If they fulfill the criteria, the respective value is updated with

```
*pmax = array[i];
```

or

```
*pmin = array[i];
```

Which at the end of the loop will have `max` containing the largest number and `min` containing the smallest number in the array. At the end of the function, we don't need to return anything, because the max and min values are already stored in variables in the main function.