Computer Programming 1 Lab

2022-10-27

andyjjrt

Outline

- Passing Array to functions
- Sorting Array
- Searching Array
- Exercise 6

Passing Array to functions

Method 1: Just pass it

```
void print2DArray(int m, int n, int array[3][4]) {
  for(int i = 0; i < m; i++) {
    for(int j= 0; j < n; j++) {
      printf("%d", array[i][j]);
    }
  }
}
```

Passing Array to functions

Method 1: Just pass it

```
void print2DArray(int m, int n, int array[][4]) {
  for(int i = 0; i < m; i++) {
    for(int j = 0; j < n; j++) {
      printf("%d", array[i][j]);
    }
  }
}

// The first subscript is not required, but all subsequent subscripts are required.
```

Passing Array to functions

Method 2: Treat it as a 1D array

```
void print2DArray(int m, int n, int array[]) {
  for(int i = 0; i < m; i++) {
    for(int j = 0; j < n; j++) {
      printf("%d", array[i * m + j]);
    }
  }
}
```

Passing Array to functions

Method 3: use malloc/calloc

```
void print2DArray(int m, int n, int** array) {
  for(int i = 0; i < m; i++) {
    for(int j = 0; j < n; j++) {
      printf("%d", array[i][j]);
    }
  }
}

int main() {
  int m, n;
  // ...
  int** array = malloc(m * sizeof(int*));
  for(int i = 0; i < m; i++) {
    array[i] = malloc(n * sizeof(int));
  }
  // ...
}
```

Sorting Array

Bubble sort (verrrrrry basic one)

```
int main() {
  // ...
  for(int i = 1; i < n; i++) {
    for(int j = 0; j < i; j++) {
      if(arr[i] < arr[j]) {
        int tmp = arr[i];
        arr[i] = arr[j];
        arr[j] = tmp;
      }
    }
  }
}
```

Searching Array

Method 1: Linear approach

```
int indexOf(int* array, int arraySize, int target) {
  for(int i = 0; i < arraySize; i++) {
    if(array[i] == target) {
      return i;
    }
  }
}
```

Searching Array

Method 2: Binary search

```
int binarySearch(int* array, int target, int low, int high) {
  int middle = (low + high) / 2;
  if(low > high) return -1; //Not found
  if(array[middle] == target) return middle;// Found in the middle
  else if(array[middle] > target) {
    return binarySearch(array, target, low, middle - 1); // Go to LHS
  }else {
    return binarySearch(array, target, middle + 1, high); // Go to RHS
  }
}
```

Exercise 6 - Matrix Multiplication

$$\begin{bmatrix} 5 & 8 & -4 \\ 6 & 9 & -5 \\ 4 & 7 & -2 \end{bmatrix} \times \begin{bmatrix} 2 \\ -3 \\ 1 \end{bmatrix} = \begin{bmatrix} -18 \\ -20 \\ -15 \end{bmatrix}$$

Any Questions?