CSC209 Week 4 Notes

Hyungmo Gu

May 13, 2020

Introduction to arrays in C 1 of 3

- Array

    - **Syntax:** $<$TYPE $>$VAR_NAME[ARRAY_SIZE]

```
1    #include <stdio.h>
2
3    int main() {
4        float daytime_high[4];
5    }
6
```

Introduction to arrays in C 2 of 3

- Accessing Array Elements

    - C doesn't check if an array access is within the bounds of array
    - Overwrites memory location if exists

```
1    #include <stdio.h>
2
3    int main() {
4        float daytime_high[4] = {1,2,3};
5        daytime_high[5] = 999;
6    }
7
```

    - Segmentation fault occurs if suitable memory location doesn't exist.

```
1    #include <stdio.h>
2
3    int main() {
4        int daytime_high[4] = {1,2,3};
5        daytime_high[3000] = 999;
6    }
7
```

Introduction to arrays in C 3 of 3

- Iterating Over Arrays

  - For loop
    * '<' is used over '<=' for the end condition,i.e. $i < 4$ in for $(i = 0; i < 4; i++)$.

```
1     #include <stdio.h>
2
3   int main() {
4       float daytime_high[4] = {16.0, 12.8, 14.6, 19.1};
5
6       float average_temp = 0;
7
8       int i;
9       for (i = 0; i < 4; i++) {
10          printf("Adding element %d with value %f.\n", index
    , daytime_high[i]);
11          average_temp += daytime_high[i];
12      }
13
14      average_temp = average_temp / 4;
15      printf("average %f\n", average_temp);
16
17      return 0;
18  }
19
```

  - Constants
    * Combines multiple repeating values into one
    * Used to increase maintainability and readibility

```
1     #include <stdio.h>
2     #define DAYS 4 // <-- HERE!!
3
4   int main() {
5       float daytime_high[DAYS] = {16.0, 12.8, 14.6, 19.1};
6
7       float average_temp = 0;
8
9       int i;
10      for (i = 0; i < DAYS; i++) {
11          printf("Adding element %d with value %f.\n", index
    , daytime_high[i]);
12          average_temp += daytime_high[i];
13      }
14
15      average_temp = average_temp / DAY;
16      printf("average %f\n", average_temp);
17
18      return 0;
19  }
20
```

Pointers in C 1 of 7

- Address in C

  - **& <VARIABLE_NAME >**
  - Returns memory location of variable

```c
#include <stdio.h>
#define DAYS 4

int main() {
    int i;
    i = 5;
    printf("Value of i: %d\n", i);
    printf("Address of i: %p\n", &i);
}
```

- Pointer

  - **<TYPE >* <VARIABLE_NAME >**
  - Is used to store memory addresses

```c
#include <stdio.h>
#define DAYS 4

int main() {
    int *pt;
    pt = &i;

    printf("value of pt: %p\n", pt);
    printf("Address of pt: %p\n", &pt);

    printf("Value pointed to by pt: %d\n", *pt);
}
```