

BA - Discussion #11, 2020-11-13

Announcements

- Don't forget to read the book! Not just the slides!
- Look at the syllabus (or calendar feature on BB) to see what material is being covered
- Project proposals due Monday! No individual submissions; make sure that all names are on the document
- Final project due the last day of class (12/9) → don't procrastinate!
- Exam #3 is three weeks from today and is just on C material
- Programming style
 - declare all variables together in functions, first thing in the main body, and then have the statements
 - no global variables ever
 - use for loop as the counted loop, and while/do while as the conditional loop
- IMPORTANT: declaring loop/iterator variables in a for loop
 - this hasn't been taught in the course, nor is it in the book, but we've decided it will be acceptable since it seems to be incorporated into modern C compilers
 - what we mean, for example:

```
for (i=0; i...)
{
    stuff
}
```

vs.

```
for (int i=0; i...)
{
    stuff
}
```

*we will accept both
- Next week will be very new concepts:
 - pointers
 - call-by-reference
 - dynamic memory allocation
- Quiz THIS AFTERNOON from 4:40 - 5:00 pm: 15 minutes to compute, 5 minutes to upload
 - alternate time zone quiz at 7 am Boston time Saturday
- When download something, such as a quiz/exam, make sure that you always compare what you download to the original... sometimes formatting is off, so make sure that you always check it!

Review of Material

- Data structures: arrays, strings, structs
 - Typedef
 - functions that return one value
 - void functions
 - program organization; function prototypes
 - anything else?
- how to use visual studio (for those you using it)

Making an Array

```
#include <stdio.h>
#define ROWS 2
#define COLS 3

int main()
{
    float arr[ROWS][COLS] = {{1, 2, 3}, {4, 5, 6}};
    int i, j;

    for (i = 0; i < ROWS; i++)
    {
        for (j = 0; j < COLS; j++)
        {
            printf("%f ", arr[i][j]);
        }
        printf("\n");
    }
    return 0;
}
```

STRINGS!!!

*don't forget the string header file, string.h *

strcpy(strvar, str);

ex.

```
strcpy(mystr1, "Hello");
printf("In the variable 'mystr1' is %s now!\n", mystr1);
```

strlen(str); ← returns the string length without the end of string sentinel

ex.

```
strlen(mystr1);
printf("In the variable 'mystr1' is %s and it is %d character long!\n", ...
      mystr1, strlen(mystr1));
```

The variable 'mystr1' is Hello and it is 5 characters long

strcat(str1, str2); ← concatenates str2 to end of str1

ex.

```
strcpy(str1, "base");
strcpy(str2, "ball");
strcat(str1, str2); ← this makes str1 "baseball" now!
```

strcmp(str1, str2); ← just compares the two strings, returning 0 if they're the same, or +/- values if not

Typedef / Structs

```
#include <stdio.h>
#include <string.h>
```

```
typedef struct{
    char id;
    float number;
} mystrtype;
```

```
int main()
{
    int vals[5] = {4, 33, 5, 2, 0};
    char myword[10] = "hockey";
    mystrtype onestruct = {'x', 123};

    printf("My word is %s\n", myword);
    printf("Its length is %d\n", strlen(myword));
    printf("vals[1] is %d\n", vals[1]);
    printf("The id is %c\n", onestruct.id);

    return 0;
}
```

My word is hockey
Its length is 6
vals[1] is 33
The id is x

Function Prototypes

```
#include <stdio.h>
```

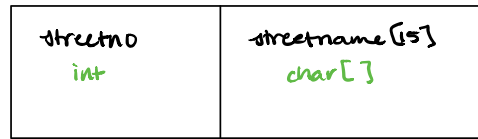
```
typedef struct
```

```
{
    int streetno;
    char streetname[15];
} street_t;
```

}



street_t (struct)



```
/* Fill in the function prototypes */
```

```
float calcstuff(int, char);
```

```
void dostuff(street_t, int*, float*);
```

↑
what mystreet looks like

avenues[20] ... 20 "rows" of the struct that
we defined as street_t

```
int main()
```

```
{
    street_t mystreet;
    avenues[20];
```

```
    float value;
```

```
    int count = 0;
```

```
    char myname[20];
```

```
    /* Assume that EVERYTHING is initialized here, */
    /* including ALL elements of all arrays and    */
    /* ALL members of all structs                  */
```

```
    value = calcstuff(mystreet.streetno, myname[0]);
```

```
    dostuff(avenues[3], &count, &value);
```

```
    return 0;
```

```
}
```

```
/* Assume that both function definitions are here */
```