CSI 3130 - Lab 3

Created	@September 29, 2022 7:22 AM
Class	CSI 3130
Type	Lab

Lab 3 - Introduction to C Programming

Install a C compiler on your machine

To run c programs on your machine, you will need to install a C compiler. If you are using your Linux system you created in the last labs, then you had already installed the Clang C complier.

You can perform this lab on any operating system. For Windows, you can install MinGW (Minimalist GNU for Windows) and for macOS you can go with Clang

You can use these tutorials to install the C compiler:

1. Windows - https://www.scaler.com/topics/c/c-compiler-for-windows/
2. macOS - https://www.scaler.com/topics/c/install-c-in-mac/

You will also need a code editor. I would suggest Visual Studio Code but you can use anything you're comfortable with.

Lab Material

Refer to the presentation in the github repo.

Programming examples

You can go through the code examples and try implementing your own functions.

```c
/*Developed by Johan Fernandes
Revised by Daniel Lobo
Course: CSI 3130*/
#include <stdio.h>
#include <stdlib.h>

//Function example 1
/*The function has two arguments both integers. The return type is also an integer.
The function requires a value and it's power and returns the result of value ^ power*/
int get_power_result(int value, int power){
  int result=1;

  for(int x1 = 0; x1 < power; x1++){
    result *= value;
  }

  return result;
}

//Function example 2
/*The implementation of the function is written at the after and otuside
the main function. This is another way to declare your functions*/
int get_factorial(int value);
float get_division(int i5, int i6);

//External variables
/*A.K.A. constant!*/
#define MAX 6

//The main function: the mothership!
int main(void) {

  //Variables
  /*You can declare and initialize here or even later in the code
  Here are data types of the integer family*/
  char c1 = 'a';
  int  i1 = 5, i2 = 10;
  const int i3 = 23;
  long l1 = 434234;

  /*Here are data types of the float family*/
  float f1 = 3.698;
  double d1 = 4545.233;

  /*We'll use this is a little later*/
  long l2 = 0;
  float f2 = 0;

  //Expressions
  printf("Expressions \n");
  l2 = l1 * i1; //Implicit type conversion from int to long
  f2 = f1 - i1; //Another implicit conversion from int to float
  //Lookout for the format specifier for long and float
  printf("Int %d, Long %ld , Float %.2f\n", i1, l2, f2);

  //Some simple arithmetic operations
```

```
    i2 = i2 - 1;
    printf("Value of i2 %d\n", i2); //Value of i2:
    //Watchout out for these guys
    printf("Value of i2 %d\n", i2++); //Value of i2:
    printf("Value of i2 %d\n", ++i2);

    //Example of constant
    /*Warning: Don't uncomment the next line. It won't work anyways. So just leave i3 be*/
    //i3++;

    printf("\n"); //Just to print a new line

    //For statement
    /*Simple for loop to print the value of the variable x1 which acts as a counter*/
    printf("For statement \n");
    for(int x1 = 0; x1 < 5; x1++){
     printf("%d \n", x1);
    }
    printf("\n");

    //Char Input and output
    printf("Char Input and output \n");
    char c2;
    /*Let's get a char from the screen. No need to input it as 'a' just simple a */
    c2 = getchar();
    /*Print the character you just got*/
    putchar(c2);

    printf("\n");

    /*A simple numerical array initilization and display with the help of a for loop*/
    printf("Numerical Arrays \n");
    //You could do it this way or the way it is shown on line 89
    int a1[MAX] = {0,1,1,2,3,5};
    // int a1[] = {0,1,1,2,3,5};
    for(int x1 = 0; x1 < MAX; x1++){
     printf("%d \n", a1[x1]);
    }

    //Functions and arguments
    /*Remember the function we created. Time to make it work*/
    int val = get_power_result(i1,3);
    printf("Power function %d \n",val);

    //Character Arrays
    /*Small experiment with character arrays*/
    printf("Input a %d letter char array \n", MAX);
    char a2[MAX];
    char c3;
    c3 = getchar(); //This is just a precaution as the  previous output has \n
                    // and this will help us remove that \n

    /*Let the user enter the text and this loop will store a word of 6 letters*/
    for( int x1=0; ((c3 = getchar()) != EOF) && (c3 != '\n') ; x1++){
      a2[x1] = c3;
    }

    //Printing the work you just extracted from the input
```

```
for (int x1=0; x1 < MAX; x1++){
  printf("%c",a2[x1]); //Time to print each character
  //putchar(a2[x1]); //This is another way to display each character
}

printf("\n");
/*Note: char arrays generally end with a \0 AKA null pointer so if you write
a string in the above input line you'll notice that the word for loop prints
the word (because the loop ends) but this print statement will print the whole
sentence will the \n
Look at the format specifier it is %s for "strings"*/
printf("%s \n",a2);

//How to find the size of the array
//printf("%ld \n", sizeof(a2)/sizeof(char));

// Loop example (do .. while)
/*This is a simple menu and will keep running till you enter 0*/
int i4=0;
do{

  printf("Select and option \n"
         "1 for Power \n"
         "2 for Factorial (by recursion) \n"
         "3 for Division \n"
         "0 to Exit \n");

  scanf("%d",&i4);

  if(i4 != 0){

    int i5, i6 = 0;

    //Switch example
    switch(i4){
      case 1:{
        printf("Enter a value and it's power (separated by space) \n");
        scanf("%d %d", &i5, &i6);
        /*You can print the return value of the function directly in the
          print statement */
        printf("Power is: %d \n", get_power_result(i5, i6));
        break;
      }
      case 2:{
        printf("Enter a value for it's factorial \n");
        scanf("%d", &i5);
        printf("Factorial is: %d \n", get_factorial(i5));
        break;
      }
      case 3:{
        /*This is an example of passing arguments as an integer but recieving a
          float value*/
        printf("Enter a dividend and divisor (separated by space) \n");
        scanf("%d %d", &i5, &i6);
        printf("Result is is: %.3f \n", get_division(i5, i6));
        break;
      }
      default:
```

```
          break;
      }

    }else{
      break; /*Kind of redudant code. There are better ways to write this logic*/
    }

  }while(i4 != 0);

  printf("Chapter 1, 2, 3 and 4 done!!");

}


//Here's the factorial function as a recursive code
int get_factorial(int value){

  if (value >= 1)
        return value * get_factorial(value - 1);
    else
        return 1;

}

//Simple division function
float get_division(int i5, int i6){
  float result = i5;

  return result / i6;
}
```