C Programming (1)

The goal of this recitation is to get started with C programming. The first three questions will help you to learn how to write C code from scratch, practice basic C syntax, data structures, and handle input/output. From the fourth question, you are going to use pointers to access memory space, be able to make functions and pass reference. The last question is an algorithm question for more advanced C programmers.

Here are some examples from class slides:

```c
#include <stdio.h>

int main()

{

    int a = 23;

    float f =0.31234;

    char str1[] = "satisfied?";

    printf("The values are %d, %f , %s\n", a, f, str1);

    scanf("%d %f",&a, &f); /* will come back to the & later */

    scanf("%s", str1);

    printf("The values are %d, %f , %s\n", a, f, str1);

}
```

Example for structure declaration:

```c
struct student_record {

        int id;

        float grade;

        };
```

Example for pointer to structure:

```c
struct student *s1;


(*s1).uid = 1007;

s1->grade = 4.5;
```

Example for pseudo-random number generation between 0 and MAX_LIMIT:

```
#include <stdlib.h>

upper = MAX_LIMIT + 1;

r = rand() % upper;
```

Preliminaries:

- After loging to the Unix machines, you should create a directory for the course ~/cs250 and consecutive subdirectories for the recitations (i.e. ~/cs250/rec1).
- Each program that you write should have a different name for its source file.
- Compile with your programs using gcc .

Useful references:

- You can web search for a linux cheat sheet and print them out as a reference, if needed.
- C reference website: http://en.cppreference.com/w/c

Task 1: "Hello World!"

Create a C program that outputs the string "*Hello World!*"

Steps:

- ssh into the teer machine
- create a source code file with nedit/gedit/vim
- write the code
- save + exit
- compile
- run

Task 2: Input/Output with different type of data

Write a program that asks the user to input a string representing his name, an integer representing his age and a float representing the current temperature. Then the program should output a sentence like *"Hello, my name is x. I am z years old. Today's temperature is t."*

Task 3: Structs

Steps:

- Declare a simple struct that represents a student's information for this class. The struct will contain 2 attributes {Duke UID, GPA}.

- Create an array of these structs and use for/while loop to fill them with arbitrary data.
  For example the UID's can be increasing numbers and the grade can be the modulo 5 of the current UID - i.e. a student with UID 1004 should have a GPA of 4, similarly a student with UID 2003 should have a GPA of 3.
  As an alternative you can generate a random GPA each time.
- The program should print 1-2 records of these structs so that you can check the correctness of your code.

Task 4: Pointers

In this section we have two separate programs:

- Create two integer pointers. Use malloc to allocate memory for the int storage. Assign two, different integer values to the int storage.
  Have the program print the address and the value of the integer for both variables.
  "Address: xxxxxx \
  Value: y"
  Lastly, free both blocks of memory using free().

- Modify the integer swap example from lecture so that the swap function accepts as arguments two addresses of student structs and swaps the values of these students. Use pointers and try to mimic the swap function for integers provided in lecture. You should declare two student struct variables in main.

Question 5 : Algorithmic (optional, if you have time...)

A prime number is a number that has exactly two divisors: itself, and 1. Using the sieve of Eratosthenes write a C program that prints the prime numbers that are less than 100.

The sieve of Eratosthenes works as follows:

1. Create a list of the numbers from 1 to 100 - lst = [1, ..., 100]
2. p ← 2 Assign to p the first prime number 2
3. Starting from p, mark all multiples of p in the lst that are greater than p (i.e. 4, 6, ...).
4. Find the first number in lst that is greater than p and is unmarked. Assign to p that value and go to step 3.