C Programming Basics

SDS 322/329

November 10, 2015

Email any questions to:
rauta@tacc.utexas.edu

TACC

THE UNIVERSITY OF TEXAS AT AUSTIN

Recap & Administrative Trivia

- Functions in C
  - String manipulation functions
  - In-class exercises

- HW-4, Part-2 & Project Assignment will be posted online soon – an announcement will be sent out

- Mid-term grades for the C part will be posted online today
  - Discussion done on it last Tuesday
  - Solutions to questions available on class slides
  - Fortran part should be available soon too

Overview of the Course

- Writing a Basic C Program
- Understanding Errors
- Comments Keywords, Identifiers, Variables
- Operators
- Standard Input and Output (Basic)
- Control Structures
- Standard Input and Output
- Arrays, Structures
- Functions in C
- Pointers
- Working with Files

All the concepts will be accompanied with examples.

Recursion (1)

- When a function calls itself it is called a recursive function (and the process is called recursion)

- In order to write a recursive function, you should be calling the function from within the function body

- The case or condition in which the recursion ends is called a base case
  - What will happen if there is no base case defined?

- The concept of recursion is similar to looping
  - Think how for-loops run for specific number of times and how they terminate
  - Like in for-loops, we must change some value incrementally to reach closer to the base case

Recursion (2)

```
void myFunction( int counter){
    if(counter == 0){
      printf("%d \n",counter);
      //other code
    }else{
        printf("%d \n",counter);
        myFunction(--counter);
        printf("%d \n",counter);
        //other code
    }
}
```

What will be the output if the function is called with counter = 10?

Source: http://www.danzig.us/cpp/recursion.html

Recursion (3)

- How will you write a function to find the factorial of an integer?
  - Example, `5!` equals `5*4*3*2*1`
  - Can we say `5!` equals `5*4!` ?
  - In seeing the factorial of `5` in this second way we have gained a valuable insight. We can now see our problem in terms of a simpler version of the problem itself , that is, we can now define the problem in terms of itself
    - `5!` in terms of `4!`
  - **This is the essence of recursive problem solving**
  - Now all we have left to do is decide what the base case is
    - What is the simplest factorial?
    - `1!`
    - `1!` equals `1`

In-Class Exercise

- Write a program to find the factorial of first `N` numbers where `N` is passed to the program by the user
  - Read the value of `N` using `scanf` or pass it as a command-line argument
  - You have to write a recursive function named `factorial` that accepts `N` as an argument

Recursive Functions: recurse.c

```
#include <stdio.h>
int fact(int n){
  int factorial;
  if(n==0){
     return 1;
  }else{
     factorial=fact(n-1)*n;
  }
  return factorial;
}
int main(){
  int num,factorial;
  printf("Enter N to find N! : ");
  scanf("%d",&num);
  if(num>=0){
      factorial=fact(num);
      printf("The factorial of %d is %d ",num,factorial);
  }
  return 0;
}
```

<-- Termination step to avoid the program from crashing

<-- Recursive call to function `fact`

A function that contains a call to itself => recursive
Such functions should have at least one exit condition that can be satisfied to avoid runtime stack overflows

In-Class Exercise - 1

- The standard library function strcpy() function can be used to copy the content of one string to another
  - Write a program that copies the content of one string to another manually without using the strcpy() function

Solution to Exercise-1

```
#include <stdio.h>
int main(){
    char s1[50], s2[50];
    int i;
    printf("Enter string s1: ");
    scanf("%[^\n]s",s1);
    for(i=0; s1[i]!='\0';i++){
        s2[i]=s1[i];
    }
    s2[i]='\0';
    printf("String s2: %s",s2);
    return 0;
}
```

References

- C Programming Language, Brian Kernighan and Dennis Ritchie
- Let Us C, Yashavant Kanetkar
- C for Dummies, Dan Gookin
- http://cplusplus.com
- https://en.wikibooks.org/wiki/C_Programming/Strings