Code Snippets

This document contains sample code to use as a reference when programming in C. Also provided are links to the API references for certain commands.

Common Libraries

stdio.h

Reference: http://www.cplusplus.com/reference/cstdio/
C library to perform Input/Output operations.

stdlib.h

Reference: http://www.cplusplus.com/reference/cstdlib/
Several general purpose functions, including dynamic memory management, random number generation, communicating with the environment, integer arithmetic, searching, sorting and converting.

math.h

Reference: http://www.cplusplus.com/reference/cmath/
Declares a set of functions to compute common mathematical operations and transformations.

time.h

Reference: http://www.cplusplus.com/reference/ctime/
This header file contains definitions of functions to get and manipulate date and time information.

Importing Headers

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

Standard Main function

```
int main(int argc, char *argv[])
{
    return 0;
}
```

Commenting

```
/*
 * Author: Llyr ap Cenydd
 * Purpose: Main method for Lab 1. All exercises for the Lab
 *          are called from this function.
 * Date : 27/1/14
 */


//single line comment
```

C Pre-Processor

Macros

```
#define MAGIC_NUM 57
```

```
#define INCREMENT(x) x++
```

```
#define MULT(x, y) (x) * (y)
int z = MULT(3 + 2, 4 + 2);
```

```
#define ADD_FIVE(a) (a) + 5
int x = ADD_FIVE(3) * 3;
```

Pausing the program

```
getchar();
system("PAUSE");
```

Declaring variables

```
int myInteger = -5;
unsigned int myInteger = 5;

short int myInteger = 50;
long int myInteger = 500000;

float myFloat = 1.23f;
```

```
double myDouble = 3.10001;

char myChar = 'c';
char myString[5] = "Hello";
```

Printing to screen (printf)

Reference:
http://www.cplusplus.com/reference/cstdio/printf/

```
printf( "Hello World\n" );
printf ( "Characters: %c %c \n " , 'a', 65);
printf ( "Decimals: %d %ld\n " , 1977, 650000L);
printf ( "Preceding with blanks: %10d \n" , 1977);
printf ( "Preceding with zeros: %010d \n" , 1977);
printf ( "Some different radices: %d %x %o %#x %#o \n " ,
100, 100, 100, 100, 100);
printf ( "floats: %4.2f %+.0e %E \n" , 3.1416, 3.1416,
3.1416);
printf ( "Width trick: %*d \n" , 5, 10);
printf ( "%s \n" , "A string" );
```

```
char myString[80] = "hello" ;
int myInt = 10;
double myFloat = 1.23;
printf ( "String variable : %s \nInt variable : %d \nFloat
variable : %f \n", myString, myInt, myFloat );
```

Read data from Keyboard (scanf)

Reference:
http://www.cplusplus.com/reference/cstdio/scanf/

```
char str [80];
int i;

printf ("Enter your family name: ");
scanf ("%79s" ,str);
printf ("Enter your age: ");
scanf ("%d" ,&i);
```

```
printf ("Mr. %s , %d years old.\n" ,str, i);
printf ("Enter a hexadecimal number: " );
scanf ("%x" ,&i);
printf ("You have entered %#x (%d).\n", i, i);
```

We can also use the fgets() function to read strings with multiple words:

```
#include <stdio.h>

char szInput[256];
printf ( "Enter a sentence: " );
fgets (szInput, 256, stdin );
printf ( "The sentence entered is %u characters long.\n"
,(unsigned)strlen(szInput));
```

Type Casting

Casting lets you convert a data type to another type, for example:

```
//cast value_i to a double
int value_i = 8;
double value_d = (double) value_i;  //cast value_i to a double
```

```
//perform a float division on two integers
int value1 = 3;
int value2 = 4;
float division = (double) value1/value2;
```

Conditional Statements

```
int value = 8;

if(value < 10)
    printf("\nValue is less than 10");
else
    printf("\nValue is greater than or equal to 10");
```

```
int char = 'c';

if(character == 'y')
    printf("\nYes");
if(character == 'n')
    printf("\nNo");
```

Random Numbers

```
#include <time.h>
#include <stdlib.h>

srand(time(NULL));  //seed with current time
int r1 = rand() % 100; //random between 0-100
int r2 = 10 + (rand() % 90); //random between 10-100
```

Strings

```
char a1[][14] = { "one", "two", "three" };
char* a2[] = { "one", "two", "three" };

printf(a1[0]); // prints one
printf(a2[0]); // prints one
printf(a2[1]); // prints two
```

Ternary operator

```
c = (c > 0)? 10 : -10;  //10 if c is > 0, otherwise -10
```

Enumerated types

```
enum colors{RED,GREEN,BLUE}

enum colors myFavouriteColor = RED;
```

```
typedef enum{monday=2, tuesday=4,wednesday=6, thursday=8,
friday=10) weekday;

weekday myFavouriteDay = friday;
```

Switch

```
int val;
printf("\nPlease enter a grade : ");
scanf("%d",&val);

switch(val / 10)
        {
        case 9:
                printf("\nGrade A*");
                break;
        case 8:
                printf("\nGrade A");
                break;
        case 7:
                printf("\nGrade B");
                break;
        case 6:
                printf("\nGrade C");
                break;
        case 5:
                printf("\nGrade D");
                break;
         default:
                 printf("\nYou failed!");
                 break;
         }
```

Dynamic 2D Array allocation

Creating a 2D array using a 1D array:

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

int main(int argc, char *argv[])
{
  int num_rows = 10;
  int num_cols = 5;
  int *array = malloc(sizeof(int) * num_rows * num_cols);

//set all elements to zero
  int x,y;
  for(x=0; x< num_rows; x++)
    for(y=0; y< num_cols; y++)
        array[(x * num_cols) + y] = 0;
}
```

Similar code, but this time using a pointer to a pointer to create an index-able 2D array.

```
int rows = 10
int cols = 20;
int i,j;

//create the rows
int  **array = (int **) malloc(sizeof(int *) * rows);

//create the columns
for(i=0; i<rows; i++)
      array[i] = (int *) malloc(sizeof(int) * cols);

//set all elements to zero
for(i=0; i<rows; i++)
      for(j=0; j<cols; j++)
            array[i][j] = 0;

//...later
free(array);
```

Copying Arrays

Assuming that the 2D array (called array) has been created and initialised as in the previous section, this code will copy the contents of array into arrayCopy. The function memcpy is found in <stdlib.h>.

```
#include <stdlib.h>
int *arrayCopy;
arrayCopy = malloc(sizeof(int[width][height]));
memcpy(arrayCopy, array, sizeof(int) * width * height);
```

Structs

```
/* this should be outside any function */

typedef struct Person {
char *forename;
char *surname;
int age;
float height;

} Person;


/* this should be inside a function */

Person person1 = {"John", "Kimble", 42, 6.0f};
Person person2 = {"Dutch", "Schaefer", 38, 6.0f};
Person person3 = {"Ian", "Malcolm", 43, 6.3f};
Person person4 = {"Guybrush", "Threepwood", 23, 6.0f};

person1.forename = "Detective";

printf("\n Forename is : %s", person1.forename);
printf("\n Surname is  : %s", person1.surname);
```

Iterating through a String

```
#include <stdio.h>
#include <stdlib.h> //needed for strlen()

      char* myString = "The quick brown fox jumps over the lazy dog";
      int i;

      for(i=0; i<strlen(myString); i++)
      {
            printf( "\nCharacter is : %c", myString[i] );
      }

    return 0;
}
```

Sentinel Controlled While Loop

```
char c;

while (c != 'q')
{
    printf("\nEnter 'q' to quit : ");
    scanf(" %c", &c);
}
```

Functions

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

//function decleration
int addition(int num1, int num2);


//main function
int main(int argc, char *argv[]) {

   int a = 5;
   int b = 10;

   int result = addition(a,b);
   printf("\nThe sum of %d and %d is %d \n", a, b, result);

   system("Pause");
   return 0;
}


//addition function
int addition(int num1, int num2)
{
   return num1 + num2;
}
```