# C Programming

**Boitumelo Phetla**

May 20, 2019

**C is a general-purpose, imperative computer programming language, supporting structured programming, lexical variable scope and recursion, while a static type system prevents many unintended operations.**

## 1 Code Snippets

### 1.1 Hello World

```
/*
* Author: Boitumelo Phetla
* How to compile on Terminal
* gcc -Wall -o hello_world hello_world.c
* ./hello_world
*/
#include <stdio.h>

int main(void){
   puts("Hello world!!"); //prints out to
        console screen
   return(100);  //returns anything
   }
```

### 1.2 printf statement

```
/*
* Author: Boitumelo Phetla
* How to compile on Terminal
* gcc -Wall -o program program.c
* ./program
*/
#include <stdio.h>

int main(void){
   printf("Hello world!!\n"); //prints out to
        console screen
   return(0);
   }
```

### 1.3 scanf statement

```
/*
* Author: Boitumelo Phetla
* How to compile on Terminal
* gcc -Wall -o program program.c
* ./program
*/
#include <stdio.h>

int main(void){
   int num = 0;  //initialization

   printf("Enter a number: ");
   scanf("%d", &num);
   printf("Number: %d\n", num);
   return(0);
   }
```

### 1.4 Simple arithmentic Algorithm

Calculate temperature in Celsius from Farenheit inputs. $C = \frac{5}{9}(F - 32)$

```
/*
temp/
      |__________celsius.h
      |__________temp.c
*/

/*celsius.h*/

#ifndef celsius_h
#define celsius_h

//C = 5/9 * (F - 32)
float cTemp(float k){
  return (9/5 * (k - 32));
}

#endif
```

```
/*temp.c*/

#include "celsius.h"
#include <stdio.h>

int main(void){

  float k[] = {100.1, 99.9, 88.8, 77.7, 66.6,
      55.5, 44.4, 33.3, 22.2, 11.1, 5.55};

  for(int i = 0; i < sizeof(k)/sizeof(int); i++){
     printf("%.2f k = %.2f C\n", k[i],
          cTemp(k[i]));
  }

  return 0;

}



/*Output*/

100.10 k = 68.10 C
99.90 k = 67.90 C
88.80 k = 56.80 C
77.70 k = 45.70 C
66.60 k = 34.60 C
55.50 k = 23.50 C
44.40 k = 12.40 C
33.30 k = 1.30 C
22.20 k = -9.80 C
11.10 k = -20.90 C
5.55 k  = -26.45 C
```

## 1.5 Preprocessor

```
#include <stdio.h>
#include <math.h> //library header file
#define PI 3.1415

//A = PI^2 * r
double area(double radius);

int main(void){
  double r = 1.00000388488484884453434343;
  printf("Area(%f) = %.2f\n", r, area(r));
  return(0);
}


double area(double radius){
  return pow(PI, 2) * radius; //math
}

/*Output*/
Area(1.000004) = 9.87
```

## 1.6 Do-While Statement

```
#include <stdio.h>
#include <math.h> //library header file
#define PI 3.1415

//A = PI^2 * r
double area(double radius);

int main(void){

  double r = 1.00000388488484884453434343;
  do{
     printf("Area(%f) = %.2f\n", r, area(r));
          //executes nonetheless
  }while(r < 1); //terminates here condition not
      met

  return(0);
}

double area(double radius){
  return pow(PI, 2) * radius; //math
}

/*Output*/
Area(1.000004) = 9.87
```

## 1.7 While statement

```
#include <stdio.h>

int main(void){

  int start = 0, stop = 100, stride = 10;
  int count = 0;
  while(start <= stop){
    printf("%d\t:\t%d\n", count, start);
    count+=1;
    start+=stride;
  }
  return 0;
}

/*Output*/

0       :      0
1       :      10
2       :      20
3       :      30
4       :      40
5       :      50
6       :      60
7       :      70
8       :      80
9       :      90
10      :      100
```

## 1.8 Constant, While, If Statement

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

#define C 299792458     //speed of light (m/s)


float e(float m);       //e = mc^2

int main(void){

  //define sentinel as m= -1
  printf("To terminate the programe enter
       [-1]\n");

  float m = 0.0;

  printf("Enter mass [kg]: ");
  scanf("%f", &m);

  while(m > 0){
      printf("m = %.2f kg, e = %.10e m/s\n", m,
           e(m));
      printf("To terminate the programe enter
           [-1]\n");
      printf("Enter mass [kg]: ");
      scanf("%f", &m);
      if(m < 0){
        printf("Program terminated\n");
      }
  }
  return 0;
}

float e(float m){
  return (m*pow(C,2));
}


/*Output*/
To terminate the programe enter [-1]
Enter mass [kg]: 20
m = 20.00 kg, e = 1.7975103338e+18 m/s
To terminate the programe enter [-1]
Enter mass [kg]: -1
Program terminated
```

## 1.9 Simple getchar putchar statements

```
#include <stdio.h>
int main(){
  char c = getchar(); //input
  putchar(c);        //display
  puts("");
  return 0;
}

/*Output*/
A
A
```

## 1.10 Array of chars

```
#include <stdio.h>

int main(){
  int c;
  c = getchar();
  while(c != EOF){ //ctrl + D or Z
    putchar(c);
    c = getchar();
  }

  return 0;
}
```

## 1.11 Main function without a type

```
#include <stdio.h>

main(){
  printf("Testing\n");
  return 0;
}

/*Output*/
main.c:3:1: warning: type specifier missing,
    defaults to 'int' [-Wimplicit-int]
main(){
^
1 warning generated.
Testing
```

## 1.12 Static variables

```
#include <stdio.h>

/*
 * Static variables have a property of preserving
 * their value even after they are out of their
     scope.
 * static data_type variable_name =
     variable_value

   static variables

   static variables are allocated memory in data
       segment, not stack segment.

   1. data segment
   2. stack segment
   3. heap segment

   static variables are initialized as 0 in
       memory.
   static variables are used to eliminate scope
       of variables or functinos.

*/
```

```
int func();

int main(void){
   for(int i = 0; i < 5; i++){
         printf("Calling static method: %d\n",
              func());
   }
   return 0;
}

int func(){
   static int count = 0;
   count++;
   return count;
}

/*Output*/
Calling static method: 1
Calling static method: 2
Calling static method: 3
Calling static method: 4
Calling static method: 5
```