Introduction to C Programming

Hong Liu
HPC Consultant
NICS
• A Brief History of C

• In 1972 C was first wrote at Bell Labs.

• In 1978 the publication of The C Programming Language caused a revolution in the computing world.

• In 1983, the American National Standards Institute (ANSI) established a committee to provide a modern, comprehensive definition of C. The resulting definition, the ANSI standard, or "ANSI C", was completed late 1988.
## Why C?

<table>
<thead>
<tr>
<th>Features of C language</th>
<th>Uses of C language:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reliability</td>
<td>Database systems</td>
</tr>
<tr>
<td>Portability</td>
<td>Graphics packages</td>
</tr>
<tr>
<td>Flexibility</td>
<td>Word processors</td>
</tr>
<tr>
<td>Interactivity</td>
<td>Spread sheets</td>
</tr>
<tr>
<td>Modularity</td>
<td>Operating system development</td>
</tr>
<tr>
<td></td>
<td>Compilers and Assemblers</td>
</tr>
<tr>
<td></td>
<td>Network drivers</td>
</tr>
<tr>
<td></td>
<td>Interpreters</td>
</tr>
</tbody>
</table>
Developing a program in a compiled language such as C requires at least four steps:

- **editing** (or writing) the program  `.c`
- **compiling** it  `.obj`
- **linking** it  `.exe`  *(stdio.h)*
- **executing** it
Using Microsoft C

- **Edit stage:**
  - Type program in using one of the Microsoft Windows editing packages.

- **Compile and link:**
  - Select `Build` from menu. `Building` option allows you to both compile and link in the same option.

- **Execute:**
  - Select the `Build` menu → then, Execute `filename.exe` menu
// Another simple mathematics calculation
#include <iostream>
#include <stdlib>
using namespace std;

int main(void)
{
    // variables declaration
    float a, b, c;
    // variables initialization
    a = 2.0;
b = 5.0;
c = b / a;
    cout << "Given a = 2.0, b = 5.0, c = b/a:"
    c = c + (a/b);
    cout << "c + (a/b): " << c << endl;
    // call the predefined function
    system("pause");
    return 0;
}

{ // TODO: Modify the Window class or styles here by modifying
  // the CREATESTRUCT cs
  return CView::PreCreateWindow(cs);
}
Unix systems

- Online terminal emulator
- [http://simpleshell.com/](http://simpleshell.com/)

- *hello world example*

- `#include <stdio.h>`
- `main()`
- `{`  
  - `printf("Hello world\n");`
- `}`
• Please note that Unix is a case sensitive operating system and files named firstprog.c and FIRSTPROG.c are treated as two separate files on these system.

• By default the Unix system compiles and links a program in one step, as follows: `cc firstprog.c`
• This command creates an executable file called a.out.

• The program is run as follows:
• `./a.out`
Add Comments to a Program

• A comment is a note to yourself (or others). All comments are ignored by the compiler.

• /* This is a comment. */

• main() /* main function*/

• {
  • printf("Hello, World! \n"); /* DisplayMessage */
  • }

Data Types

You can create *variables* to store *values* in. There are five basic data types associated with variables:

- **int** - integer: a whole number.
- **float** - floating point value: ie a number with a fractional part.
- **double** - a double-precision floating point value.
- **char** - a single character.
- **void** - valueless.
• An `int` variable can store a value in the range -32768 to +32767. No fractional part is allowed.

• To declare an `int`: `int variable name;`

• `int a ;` Declares that you want to create an `int` variable called `a`.

• To assign a value to our integer variable we would use the following C statement:

• `a=10;`
• Decimal Number Variables: float and double.

• float: A float number has about seven digits of precision and a range of about 1.E-36 to 1.E+36.

• double: A double number has about 13 digits of precision and a range of about 1.E-303 to 1.E+303.

• To declare: float total; double sum;

• Assign a numerical value to our floating point and double precision variables:

  • total=0.0; sum=12.50;
• **Character Variables**

• To declare a variable of type character we use the keyword `char`.

• For example:

• `char c;`

• To assign, or store, a character value in a `char`:

• `c='A'`
void

• Basically it means "nothing" or "no type"

• In C if you don’t specify the return type, the compiler automatically inferred that you wanted to return an int

• Function return value: void myFunc(int) -- the function returns nothing
True and False in C

- In C `true` is represented by any numeric value not equal to 0 and `false` is represented by 0.

- `if(a)`
  - If `a` isn't zero then this also acts as the value `true`
Mathematical operations

Add, subtract, multiply and divide.

• add   \( a + b \)
• subtract   \( a - b \)
• multiply   \( a \times b \)
• divide   \( a/b \)
• What is the answer to this simple calculation?
• \( a = \frac{10}{3} \)
• The answer depends upon how \( a \) was declared. If it was declared as type \texttt{int} the answer will be 3; if \( a \) is of type \texttt{float} then the answer will be 3.333.
• #include <stdio.h>
• main(){
  • int a,b,average;
  • a=10; b=6;
  • average = ( a+b ) / 2 ;
  • printf("Here is the answers.. ");
  • printf("\n");
  • printf("%d.",average);
  • printf("\n");
• }


Input and Output Functions

• Input functions, called `scanf`
  • `scanf("%d",&a);`

• Output functions, called `printf`
  • `printf("The value stored in a is %d",a);`
#include <stdio.h>

main()
{
    int a,b,c;
    printf("\n The first number is ");
    scanf("%d",&a);
    printf("The second number is ");
    scanf("%d",&b);
    c=a+b;
    printf("The answer is %d \n",c);
}
• `#include <stdio.h>`
• `main()`
• {
  • int dec = 5;
  • char ch = 's';
  • float pi = 3.14;
  • `printf("%d %f %c\n", dec, pi, ch);`
}
Functions

A function has the general form:

```c
#include<stdio.h>
void demo()
{
    printf("Hello");
}
main()
{
    demo();
}
```

type FunctionName (type declared parameter list)
{
    statements that make up the function
}
Making The Connections

• How to get data into a function? *parameters* are used to carry data values into a function. Parameters are listed and declared in between the () brackets in the function's definition

```c
• sum( int a, int b){
  • int result;
  • result=a + b;
  • }
• sum(1,2);
```

• How do we get values out?
• return *value*;
• int sum(int a, int b) {
  • int result;
  • result = a + b;
  • return result;
• }
• And to use it you can write something like:
• r = sum(1, 2);

• void demo();
• is a function with no parameters and no return value.
```c
#include<stdio.h>

int sum(int a, int b){
    int result;
    result=a+b;
    return result;
}

main(){
    int r; 
    r=sum(1,3);
    printf("The answer is %d.\n", r);
}
```
The Standard Library Functions

- **stdio.h: I/O functions:**
  - `printf()` as previously described
  - `scanf()` as previously described
- **string.h: String functions**
  - `strcpy()` copies contents of str2 to str1
- **ctype.h: Character functions**
  - `islower()` returns non-0 if arg is lowercase letter
  - `isupper()` returns non-0 if arg is uppercase letter
- **math.h: Mathematics functions**
  - `sqrt()` returns square root of num
- **time.h: Time and Date functions**
  - `time()` returns current calendar time of system
Data Types Part II

- So far we have looked at *local* variable now we switch our attention to other types of variables supported by the C programming language:
  - **Global variables**
  - `int max;`
  - `main(){`
  - .....`
  - `}`
  - `f1(){`
  - .....`
  - `}`
• **Constant Data Types**
  
  • Fixed values that may not be altered by program
    • #define CONSTANTNAME value
    • For example:
      • #define SALESTAX 0.05

  • #define SALESTAX 0.05
  • #include <stdio.h>
  • main() {
    • float amount, taxes, total;
    • printf("Enter the amount purchased : ");
    • scanf("%f",&amount);
    • taxes = SALESTAX*amount;
    • printf("The sales tax is £%4.2f",taxes);
    • printf("\n The total bill is £%5.2f",total);
  • }

Questions?