# 第02章 源程序 TinyC

TinyC 只用到了 C 语言中非常小的一部分，是 C 语言中非常小的子集，所有 C 语法的规则均适用于 TinyC 语法， TinyC 源程序可直接用 gcc 编译。 C 语法本书不介绍，仅介绍 TinyC 特有的部分。

## 2.1 数据类型及源程序结构

TinyC 中变量只有 int 一种数据类型（32位），函数的返回值可以声明为 int 和 void 两种类型，但编译器会自动为 void 函数返回一个 int 值。不支持全局变量，只有局部变量，变量须先声明再使用，且变量声明必须放在函数体的最前面，不支持声明变量的时候赋初值。

不支持函数原型声明，函数声明必须和定义在一起，函数无需先定义再使用。整个程序必须有一个不带参数的 **main** 函数，此为程序的入口。

“// ...” 以及 “# ...” 为单行注释。不支持 #include 等预处理命令，不支持多行注释。

典型的 TinyC 源程序是由一个个的函数定义组成的，如下：

```
int main() {
    int a, b;
    int c, d;    // 变量声明必须放在函数体的最前面
    a = 0;
    ...
}

void func1(int a, int b) {
    ...
}

...
```

TinyC 函数体内的语句只有四种：赋值语句、函数调用语句、控制语句（ if 语句）和循环语句（ while 语句）。赋值语句中，左边为变量名，右边为表达式，一个只含有表达式（函数调用除外）的语句是不合法的，如下：

```
a = 1 + a;                  // 合法
sum(1, 2);                  // 合法
if (a > 0) { ... }          // 合法
while (a < 0) { ... }       // 合法
1;                          // 不合法
1 + 2;                      // 不合法
```

## 2.2 数据运算

TinyC 支持以下算术、比较和逻辑运算：

```
+, -, *, /, %, ==, !=, >, <, >=, <=, &&, ||, !, -
```

注意上面最后一个“-”表示“反号”，应和“减号”区别开来。

TinyC 不支持 ++ 和 - - 。赋值语句只能单独使用，不能放在表达式内部，如：

```
x = y = 1;      // 不合法
(x = 1) > 0;    // 也不合法
```

## 2.3 输入及输出

TinyC 提供两个基本的 io 命令， print 和 readint 。如：

```
print("x = %d, y = %d", 2, 3);          // 输出： x = 2, y = 3
x = readint("Please input an integer");
```

print 命令将字符串打印至标准输出，并自动换行，仅支持 %d 格式化。readint 命令先打印提示信息，再从标准输入中读取一个整数并返回。注意，readint 命令必须放在赋值语句的右边，单独的 readint 命令是不合法的，而 print 命令只能单独使用，不能放在赋值语句的右边：

```
x = readint("Please input an integer"); // 合法
readint("Please input an integer");     // 不合法
print("x = %d, y = %d", 2, 3);          // 合法
x = print("x = %d, y = %d", 2, 3);      // 不合法
```

## 2.4 控制及循环语句

TinyC 仅支持 if/else 和 while 语句， while 循环中支持 continue 和 break 。不支持 for 、 switch 、 goto 等其他语句。if/else 和 while 的执行体必须用花括号括起来。如：

```
if (x > 0) y = 1;       // 不合法
if (x > 0) { y = 1; }   // 合法
```

## 2.5 函数调用

TinyC 支持函数调用，支持递归。

## 2.6 关键字

TinyC 中的关键字只有下面这些：

```
void, int, while, if, else, return, break, continue, print, readint
```

## 2.7 典型 TinyC 程序

好了，以上就是 TinyC 的全部了，够简单吧。典型的 TinyC 程序如下：

```
#include "for_gcc_build.hh" // only for gcc, TinyC will ignore it.

int main() {
    int i;
    i = 0;
    while (i < 10) {
        i = i + 1;
        if (i == 3 || i == 5) {
            continue;
        }
        if (i == 8) {
            break;
        }
        print("%d! = %d", i, factor(i));
    }
    return 0;
```

```
}

int factor(int n) {
    if (n < 2) {
        return 1;
    }
    return n * factor(n - 1);
}
```

---

以上代码中的第一行的 #include “for_gcc_build.hh” 是为了利用gcc来编译该文件的，TinyC 编译器会注释掉该行。`for_gcc_build.hh` 文件源码如下：

---

```
#include <stdio.h>
#include <string.h>
#include <stdarg.h>

void print(char *format, ...) {
    va_list args;
    va_start(args, format);
    vprintf(format, args);
    va_end(args);
    puts("");
}

int readint(char *prompt) {
    int i;
    printf(prompt);
    scanf("%d", &i);
    return i;
}

#define auto
#define short
#define long
#define float
#define double
#define char
#define struct
#define union
#define enum
#define typedef
#define const
#define unsigned
#define signed
```

```
#define extern
#define register
#define static
#define volatile
#define switch
#define case
#define for
#define do
#define goto
#define default
#define sizeof
```

---

此文件中提供了 print 和 readint 函数，另外，将所有 C 语言支持、但 TinyC 不支持的关键词全部 define 成空名称，这样来保证 gcc 和 TinyC 编译器的效果差不多。利用 gcc 编译的目的是为了测试和对比 TinyC 编译器的编译结果。

让我们先用 gcc 编译并运行一下上面这个典型的 TinyC 源文件吧。将以上代码分别存为 `tinyc.c` 和 `for_gcc_build.hh`，放在同一目录下，打开终端并 cd 到该目录，输入：

```
$ gcc -o tinyc tinyc.c
$ ./tinyc
```

将输出：

```
1! = 1
2! = 2
4! = 24
6! = 720
7! = 5040
```

如果您的系统中没有 gcc ，则应先安装 gcc 。如果你使用的是 debian ，可以用 apt-get 命令来安装，如下：

```
$ sudo apt-get install build-essential
```

**第 2 章完**