## 实验二：C编程环境

### 实验目的

1． 熟悉Linux下C程序设计的环境；

2． 对系统调用有初步了解。

### 实验时间

3学时

### 实验内容

1．Linux下C语言程序的开发过程

a、在用户主目录下用vi编辑C语言源程序（源程序已附后），如：$vi hello.c。

b、用gcc编译C语言源程序：$gcc ./hello.c -o example

这里gcc是Linux下的C语言程序编译器（GNU C Compiler），./hello.c表示待编译的源文件是当前工作目录下的hello.c，-o example表示编译后产生的目标代码文件名为example。

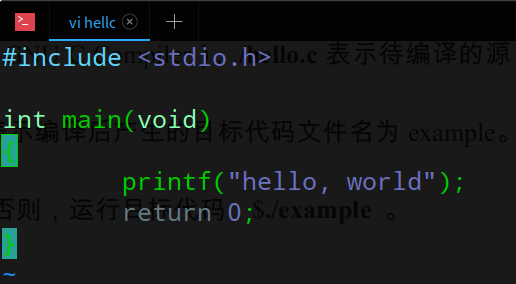
c、若编译不正确，则进入vi修改源程序，否则，运行目标代码：$./example 。

注意：

a、这只是gcc最基本的用法，其他常用选项有：-c , -S , -O , -O2, -g 等。

b、调试程序可以用gdb（GNU debugger）。

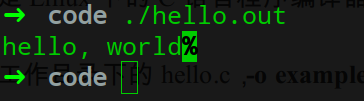
答：a、使用vi编辑C语言源程序：



b、用gcc编译C语言源程序：



c、运行目标代码：



需要注意的是输出结果中的%是我使用的zsh用于标记输出内容结尾没有换行的符号，不是程序的输出内容。

2．编辑、调试下面c语言程序，说明该程序的功能。

#include <stdio.h>

main() {

int n,a[200],carry,temp,i,j,digit = 1;

printf("Please input n:");

scanf("%d",&n);

a[0] = 1;

for( i = 2; i <= n; ++i) {

for( j = 1, carry = 0; j <= digit; ++j) {

temp = a[j-1] \* i + carry;

a[j-1] = temp % 10;

carry = temp / 10; }

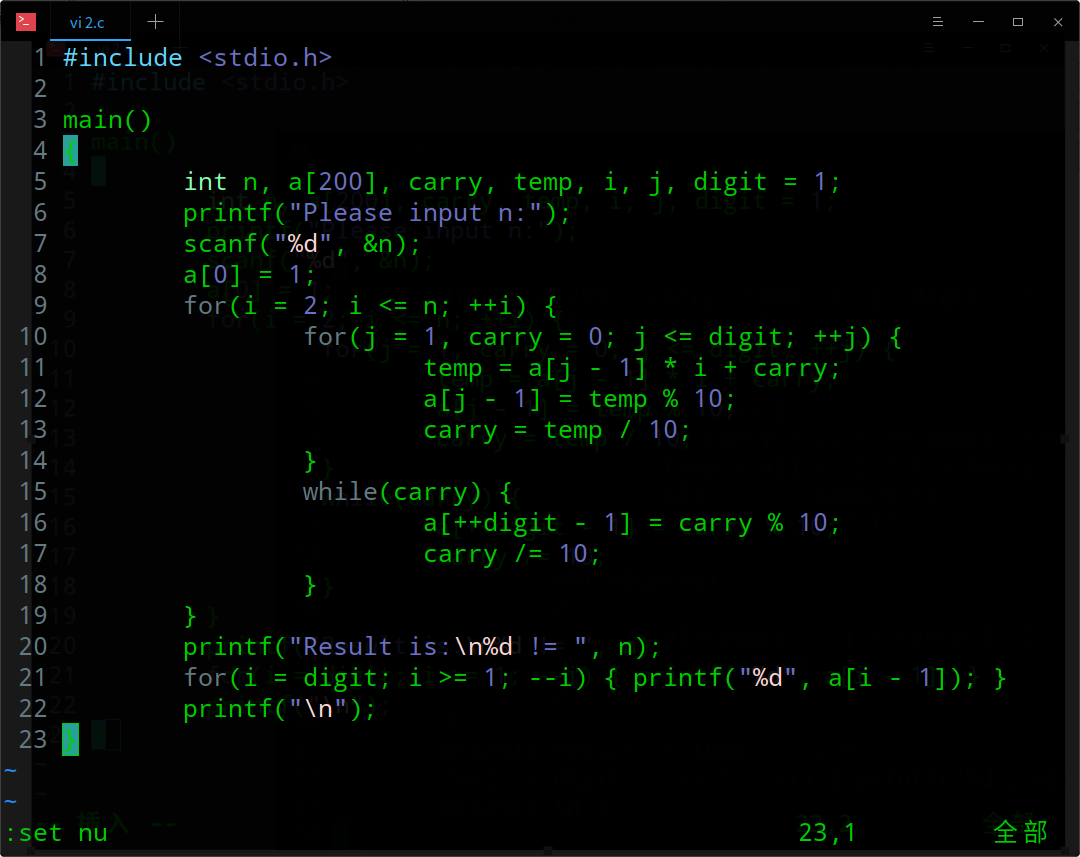
while(carry) { a[++digit-1] = carry % 10; carry /= 10;

} }

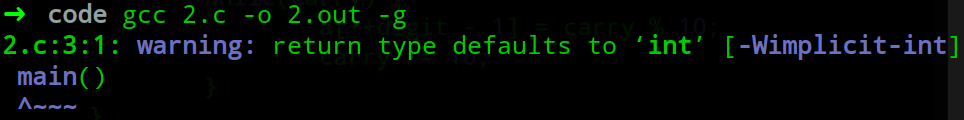
printf("Result is:\n%d ! = ",n);

for( i = digit; i >=1; --i) { printf("%d",a[i-1]); } printf("\n"); }

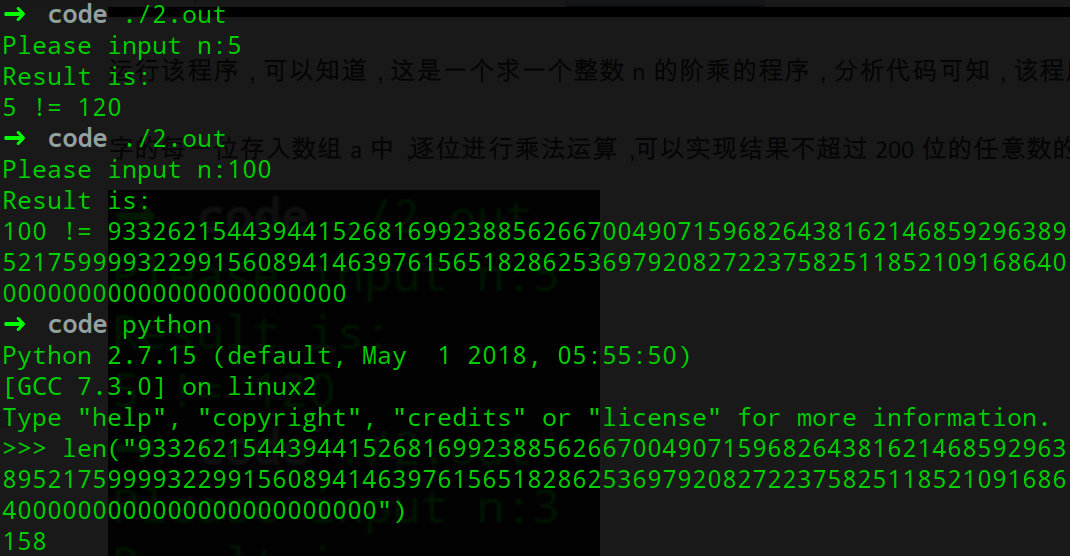
答：使用vi编辑程序如下：



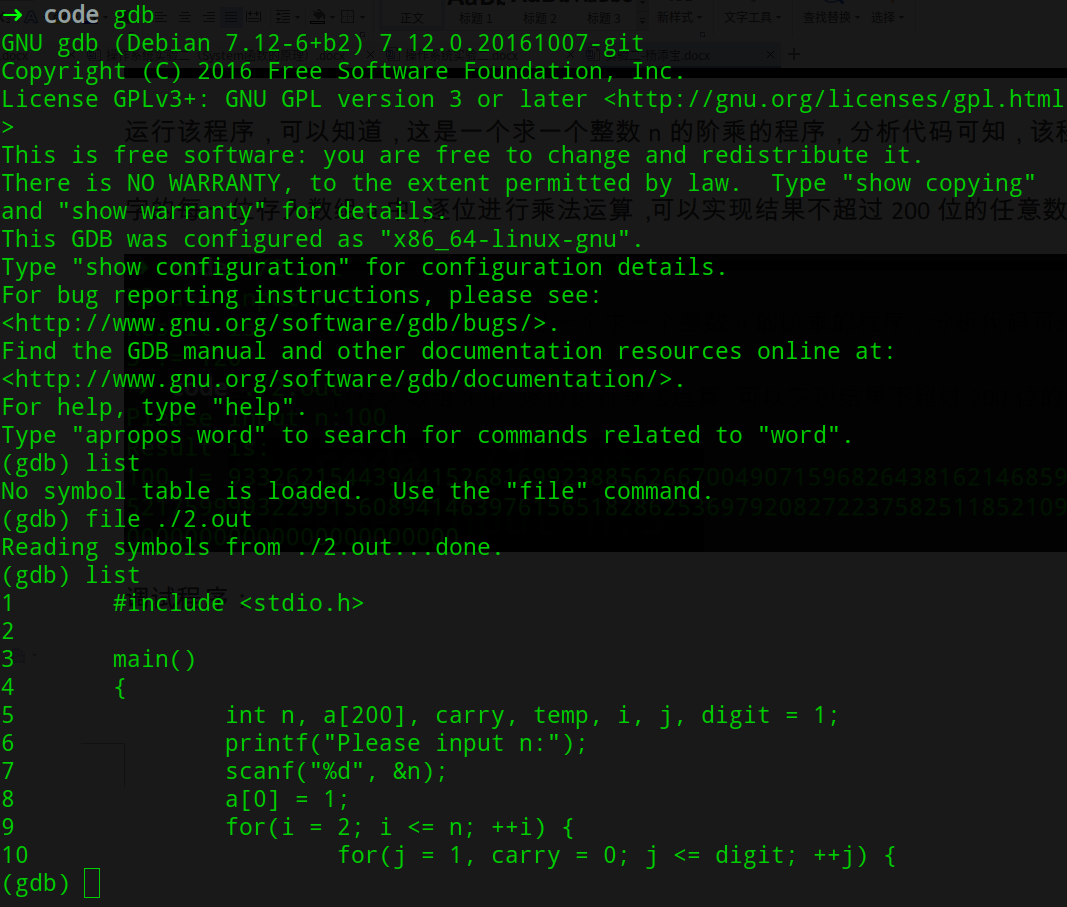
使用-o参数指定编译后的文件名，-g参数用于产生用于gdb的调试信息，编译程序如下：



运行该程序，可以知道，这是一个求一个整数n的阶乘的程序，分析代码可知，该程序对数字的每一位存入数组a中，逐位进行乘法运算，可以实现结果不超过200位的任意数的阶乘。运行结果如下图所示：



使用gdb调试程序如下图所示：



3． 编写命令解释程序

（1）内容：

利用C语言编写一个微型命令解释程序，接受并解释以下命令：

1. dir //列当前目录
2. cop 文件1 文件2 //拷贝文件
3. era文件名 //删除文件
4. dis字符串 //显示字符串
5. end //结束，退出

（2）要求：

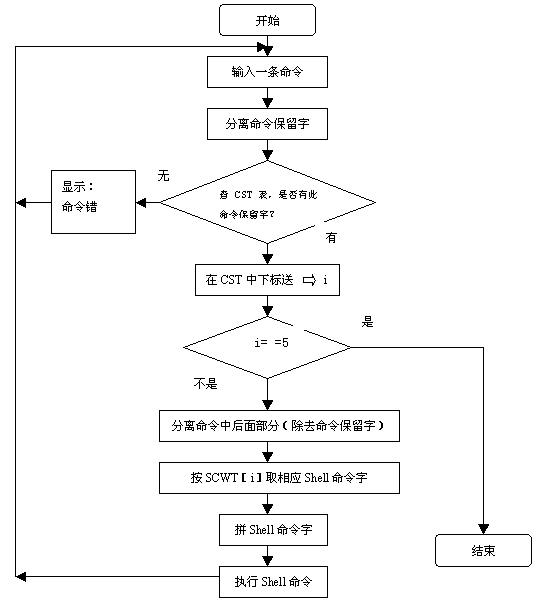
1. 命令应该由空格隔开；
2. 进行命令合法性检查，若不合法，显示出错信息，等待重新输入；
3. 调用shell命令来完成各项功能。

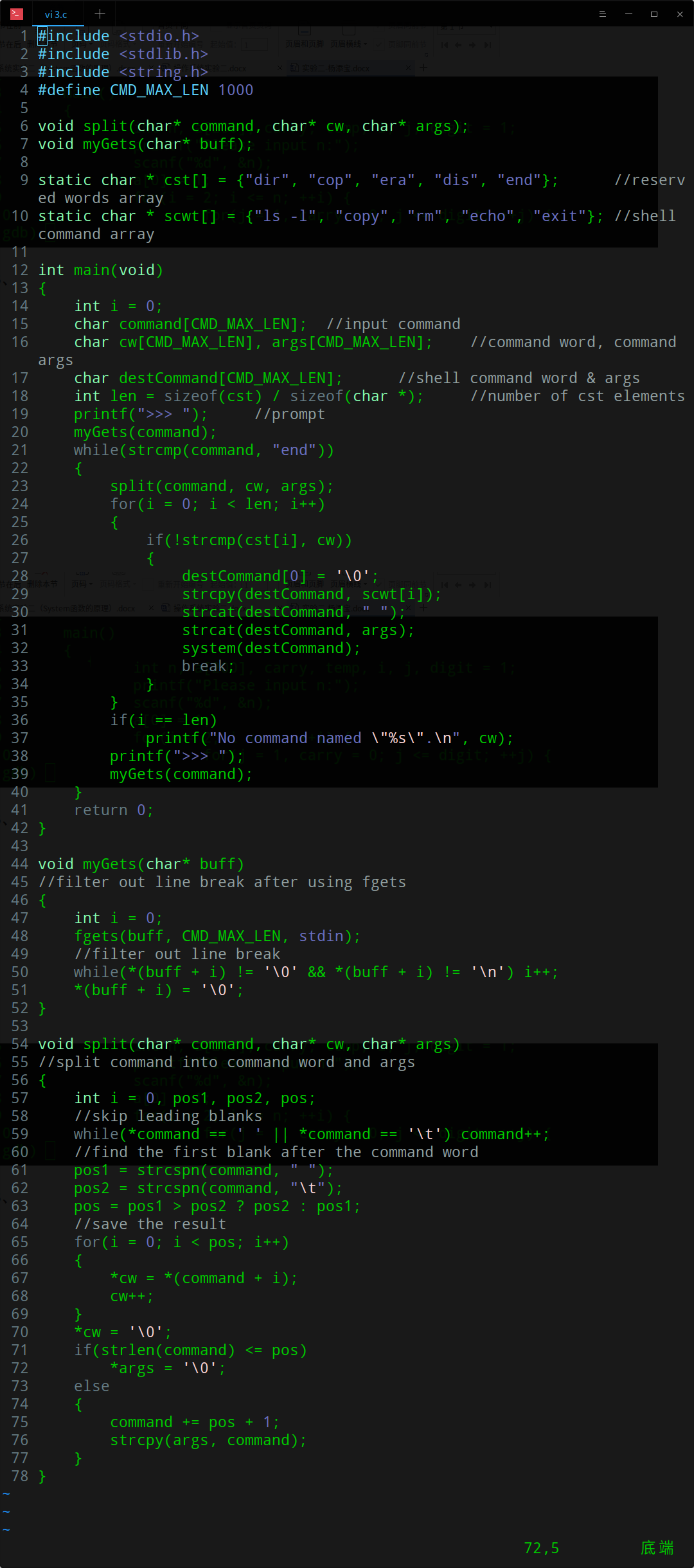
（3）思路：(不必拘泥与此，根据自己的理解和想法去编程。)

1. 用静态指针数组或二维数组形式定义命令保留字表和shell命令字表。  
    静态数组形式如下：  
    static char \* cst [ ]="dir"…"end"；static char \* scwt [ ]="ls —l"…"exit"；
2. 输入命令字gets(string)；
3. 分离命令字strcspn()；
4. 比较命令字strcmp()；
5. 执行shell命令：使用系统调用system()。

（4）命令解释程序模拟算法流程图示例

使用vi编辑的程序截图如下：





程序代码如下：

|  |  |
| --- | --- |
| **1**  **2**  **3**  **4**  **5**  **6**  **7**  **8**  **9**  **10**  **11**  **12**  **13**  **14**  **15**  **16**  **17**  **18**  **19**  **20**  **21**  **22**  **23**  **24**  **25**  **26**  **27**  **28**  **29**  **30**  **31**  **32**  **33**  **34**  **35**  **36**  **37**  **38**  **39**  **40**  **41**  **42**  **43**  **44**  **45**  **46**  **47**  **48**  **49**  **50**  **51**  **52**  **53**  **54**  **55**  **56**  **57**  **58**  **59**  **60**  **61**  **62**  **63**  **64**  **65**  **66**  **67**  **68**  **69**  **70**  **71**  **72**  **73**  **74**  **75**  **76**  **77**  **78** | #include <stdio.h>  #include <stdlib.h>  #include <string.h>  #define CMD\_MAX\_LEN 1000  void split(char\* command, char\* cw, char\* args);  void myGets(char\* buff);  static char \* cst[] = {"dir", "cop", "era", "dis", "end"};  //reserved words array  static char \* scwt[] = {"ls -l", "cp", "rm", "echo", "exit"};   //shell command array  int main(void)  {  int i = 0;  char command[CMD\_MAX\_LEN];  //input command  char cw[CMD\_MAX\_LEN], args[CMD\_MAX\_LEN];    //command word, command args  char destCommand[CMD\_MAX\_LEN];  //shell command word & args  int len = sizeof(cst) / sizeof(char \*); //number of cst elements  printf(">>> "); //prompt  myGets(command);  while(strcmp(command, "end"))  {  split(command, cw, args);  for(i = 0; i < len; i++)  {  if(!strcmp(cst[i], cw))  {  destCommand[0] = '\0';  strcpy(destCommand, scwt[i]);  strcat(destCommand, " ");  strcat(destCommand, args);  system(destCommand);  break;  }  }  if(i == len)  printf("No command named \"%s\".\n", cw);  printf(">>> ");  myGets(command);  }  return 0;  }  void myGets(char\* buff)  //filter out line break after using fgets  {  int i = 0;  fgets(buff, CMD\_MAX\_LEN, stdin);  //filter out line break  while(\*(buff + i) != '\0' && \*(buff + i) != '\n') i++;  \*(buff + i) = '\0';  }  void split(char\* command, char\* cw, char\* args)  //split command into command word and args  {  int i = 0, pos1, pos2, pos;  //skip leading blanks  while(\*command == ' ' || \*command == '\t') command++;  //find the first blank after the command word  pos1 = strcspn(command, " ");  pos2 = strcspn(command, "\t");  pos = pos1 > pos2 ? pos2 : pos1;  //save the result  for(i = 0; i < pos; i++)  {  \*cw = \*(command + i);  cw++;  }  \*cw = '\0';  if(strlen(command) <= pos)  \*args = '\0';  else  {  command += pos + 1;  strcpy(args, command);  }  } |

上述程序中包含了3个函数，主函数main控制整个程序的输入，在输入end时退出，使用myGets函数得到一行内容，使用myGets的原因是gets函数已被弃用，myGets函数内部通过调用fgets函数得到一行内容，fgets得到的内容包含末尾的换行符，myGets函数负责去掉换行符。然后再通过调用split函数将命令与参数列表分离，支持使用水平制表符或空格（1个或多个）分割命令与参数。

执行效果演示：

