**SmallC编译器**

**软件测试说明书**

**作者：吕碧茹**

**学号：10122130102**

**日期：12/16/2015**

1. **概述**

|  |  |
| --- | --- |
| **测试描述** | 对编译器的功能进行测试，主要测试其是否具有词法分析、语法分析、出错处理、生成代码等功能。 |
| **测试环境** | PC机、Windows系统、VS2010 |

1. **测试用例描述**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Test No. | No.1 | |  |  |  |  |  |
| 测试目标 | | 程序中存在错误，能够指出行号、错误号和错误类型 | | | | | |
| 测试用例的源程序： | | void main()  {  int a,b,c,d,  int e;  a=2;  while(a<100  {  b=2;  c=0;  e=a/2;  while b<=e)  {  d=a%b;  if(d==0)  {c=1;}  b=b+1;  }  if(c==0)  {write a;}  a=a+1;  }  }. | | | | | |
| 测试步骤 | | 打开文件->编译 | | | | | |
| 测试结果 | | 3：error4:这里必须是一个标识符  3：error5:丢了一个分号(或逗号)  6：error22:丢了右括号  10：error23:丢了左括号  10：error19:该语句跟着一个不正确的使用符号  10：error19:该语句跟着一个不正确的使用符号  Errors in smallC program! | | | | | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Test No. | No.1 | |  |  |  |  |  |
| 测试目标 | | 判断编译器能否成功编译程序并输出1-100之间的素数 | | | | | |
| 测试用例的源程序： | | void main()  {  int a,b,c,d,e;  a=2;  while(a<100)  {  b=2;  c=0;  e=a/2;  while(b<=e)  {  d=a%b;  if(d==0)  {c=1;}  b=b+1;  }  if(c==0)  {write a;}  a=a+1;  }  }. | | | | | |
| 测试步骤 | | 打开->编译->执行 | | | | | |
| 测试结果 | | 编译器能够编译程序输出1-100之间的素数。 | | | | | |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Test No. |  | No.2 |  |  |  |  |
| 测试目标 | | 判断编译器是否能够求出两个整数的最小公倍数 | | | | |
| 测试用例的源程序： | | void opt(int a, int b)  {  int c,d,e;  d=a;  e=b;  while(a!=b)  {  if(a>b)  {  c=a;  a=b;  b=c;  }  b=b-a;  }  c=a;  a=d/c;  b=e/c;  c=a\*b\*c;  write c;  }  void main()  {  int a,b;  read a;  read b;  call opt(a,b);  }. | | | | |
| 测试步骤 | | 打开->编译->运行 | | | | |
| 测试结果 | | 能正确求两个整数的最小公倍数。 | | | | |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Test No. |  |  | No.3 |  |  |  |
| 测试目标 | | 能否求一个整数的阶乘 | | | | |
| 测试用例的源程序： | | void main()  {  int a,b;  b=1;  read a;  while(a>0)  {  b=b\*a;  a--;  }  write b;  }. | | | | |
| 测试步骤 | | 打开->编译->运行 | | | | |
| 测试结果 | | 能求一个整数的阶乘。 | | | | |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Test No. |  |  |  | No.4 |  |  |
| 测试目标 | | Odd测试 | | | | |
| 测试用例的源程序： | | void main()  {  int a,b;  b=1;  read a;  if(odd a)  {  b=b\*a;  a--;  }  write b;  }. | | | | |
| 测试步骤 | | 打开->编译->运行 | | | | |
| 测试结果 | | 编译器能够识别odd并正确判断 | | | | |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Test No. |  |  |  |  | No.5 |  |
| 测试目标 | | For语句测试 | | | | |
| 测试用例的源程序： | | void main()  {  int a,b,n;  b=1;  for (a=10;a>0;a--;)  {  b=b\*a;  }  write b;  }. | | | | |
| 测试步骤 | | 打开->编译->运行 | | | | |
| 测试结果 | | 编译器能够识别并正确编译for语句 | | | | |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Test No. |  |  |  |  |  | No.6 |
| 测试目标 | | 异或运算xor测试 | | | | |
| 测试用例的源程序： | | void main()  {  int a,b;  b=1;  read a;  b=b xor a;  write b;  }. | | | | |
| 测试步骤 | | 打开->编译->运行 | | | | |
| 测试结果 | | 编译器能够识别并编译xor符号，输出正确结果。 | | | | |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Test No. |  |  |  |  |  | No.7 |
| 测试目标 | | Switch,case,break语句测试 | | | | |
| 测试用例的源程序： | | void main()  {  int a,b,c,d,e,f,g;  repeat  {  read a;  switch (a)  case 2 :  b=7;  c=3;  d=b%c;  write d;  d=b^c;  write d;  d++;  write d;  d--;  write d;  break;  case 3:  continue;  break;  case 4:  exit;  break;  }  until(a!=1)  }. | | | | |
| 测试步骤 | | 打开->编译->运行 | | | | |
| 测试结果 | | 编译器能够识别并编译switch，case和break语句，输出正确结果。 | | | | |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Test No. |  |  |  |  |  | No.8 |
| 测试目标 | | Repeat，continue语句测试 | | | | |
| 测试用例的源程序： | | void main()  {  int a;  a=10;  repeat  {  write a;  a--;  continue;  a--;  }  until (a>0)  }. | | | | |
| 测试步骤 | | 打开->编译->运行 | | | | |
| 测试结果 | | 编译器能够识别并编译continue，repeat，until语句，输出正确结果。 | | | | |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Test No. |  |  |  |  |  | No.9 |
| 测试目标 | | Exit，call语句测试 | | | | |
| 测试用例的源程序： | | void add()  {  eixt;  }  void main()  {  call add;  } | | | | |
| 测试步骤 | | 打开->编译->运行 | | | | |
| 测试结果 | | 编译器能够识别并编译exit，call语句，输出正确结果。 | | | | |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Test No. |  |  |  |  |  | No.10 |
| 测试目标 | | Const定义测试 | | | | |
| 测试用例的源程序： | | void main()  {  const a=3;  int b;  int c;    b=2;  c=a+b;  write c;  }. | | | | |
| 测试步骤 | | 打开->编译->运行 | | | | |
| 测试结果 | | 编译器能够识别并编译const符号，输出正确结果。 | | | | |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Test No. |  |  |  |  |  | No.11 |
| 测试目标 | | 嵌套调用call测试 | | | | |
| 测试用例的源程序： | | void daa()  {  exit;  }  void add()  {  call daa;  }  void main()  {  call add;  }. | | | | |
| 测试步骤 | | 打开->编译->运行 | | | | |
| 测试结果 | | 编译器能够识别并编译连续嵌套的call语句，输出正确结果。 | | | | |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Test No. |  |  |  |  |  | No.12 |
| 测试目标 | | 带参数的函数调用的测试 | | | | |
| 测试用例的源程序： | | void add (int a, int b)  {  int c;  c=a+b;  return c;  }  void main()  {  int a,b;  a=1; b=2;  a=call add(a,b);  write a;  }. | | | | |
| 测试步骤 | | 打开->编译->运行 | | | | |
| 测试结果 | | 编译器能够识别并正确编译带参数的函数定义，输出正确结果。 | | | | |