编译原理第一次实验测试用例：目录

1 A 组测试用例

本组测试用例共 5 个，均为比较简单的程序，分别检查针对赋值 -算数语句、分支语句、循环、数组表达式和函数调用的翻译。

1.1 A-1

待翻译程序

```
int main()
{
    int a = 10, b = 20, c = 3;
        int d = a + b *c -b;
        write(d);
        d = (a+b) / c;
        write(d);
        a = a + b;
        b = a - b;
        a = a - b;
        b = (a-b)/2 + (a/b)*2;
        write(b);
        return 0;
}
```

程序输入：无输入；预期输出：50 10 9

说明：这个用例针对赋值和算术语句进行检测。预期中的每个数字会占一行，这里为了节省中间写在同一行（下同）。

1.2 A-2

待翻译程序

```
int main()
{
        int a = 0, b = 1;
        int c = read();
        if(c < 5)
```

```
6                   write(a);
7           else
8                   write(b);
9           return 0;
10  }
```

程序输入：4；预期输出：0

程序输入：8；预期输出：1

说明：这个用例针对分支语句进行测试。

1.3 A-3

待翻译程序

```
1   int main()
2   {
3           int i = 20;
4           while(i>5) {
5                   write(i);
6                   i = i-1;
7           }
8           return 0;
9   }
```

程序输入：无；预期输出：20 19 18 $\cdots$ 6

说明：这个用例测试针对循环的翻译。

1.4 A-4

待翻译程序

```
1   int main()
2   {
3       int a[3];
4       int b;
5       a[0] = 5;
6           a[1] = 8;
```

```
        a[2] = 6;
        b = a[0] + a[1] + a[2];
        write(b);
        return 0;
}
```

程序输入：无；预期输出：19

说明：这个用例测试针对数组的翻译。

1.5 A-5

待翻译程序

```
int inc(int x)
{
        return x + 1;
}

int main()
{
        int a = 10;
        a = inc(a);
        write(a);
        return 0;
}
```

程序输入：无；预期输出：11

说明：这个用例测试针对函数调用的翻译。

2 B 组测试用例

本组测试用例共 4 个，程序较 A 类测试用例相比略为复杂。这里不专门针对赋值和算术语句设计测试用例。

2.1 B-1

待翻译程序

```
int main()
{
    int year = read();
    if(year <= 0)
        write(-1);
    else {
        int remain4 = year - (year/4)*4;
        int remain100 = year - (year/100)*100;
        int remain400 = year - (year/400)*400;
        if(remain4 != 0)
            write(0);
        else if(remain400 == 0)
            write(1);
        else if(remain100 == 0)
            write(0);
        else
            write(1);
    }
    return 0;
}
```

程序输入：1996；预期输出：1

程序输入：2000；预期输出：1

程序输入：1000；预期输出：0

程序输入：1001；预期输出：0

程序输入：0；预期输出：-1

说明：这个用例测试针对分支的翻译，程序输入一个年份，判断这个年份是不是闰年。

2.2 B-2

待翻译程序

```
int main()
{
```

```
3           int n = read();
4           int i = 2;
5           while(i<=n) {
6                   int r = 2;
7                   int half = i / 2;
8                   int isPrime = 1;
9                   while(r <= half) {
10                          int remainr = i - i / r * r;
11                          if(remainr == 0)
12                                  isPrime = 0;
13                          r = r + 1;
14                  }
15                  if(isPrime == 1)
16                          write(i);
17                  i = i+1;
18          }
19          return 0;
20  }
```

程序输入：20；预期输出：2 3 5 7 11 13 17 19

说明：这个用例测试针对循环语句的翻译，程序输入一个正整数 n，并输出不大于 n 的所有素数。

2.3 B-3

待翻译程序

```
1   int main()
2   {
3           int a[10];
4           int i = 0, j = 10;
5       int min = 0;
6           while(i<j) {
7                   int ti = (i+1) * (i+1) * (i+1);
8                   a[i] = ti - ti/j*j;
```

```
                i = i+1;
        }
        i = 1;
        while(i<j) {
                if(a[i] < a[min]) {
                        min = i;
                }
                i = i+1;
        }

        write(a[min]);
        return 0;
}
```

程序输入：无；预期输出：0

说明：这个用例主要测试针对数组的翻译，程序中产生一个伪随机的数组，然后找出数组的最小值。这里也涉及到了循环和分支。

2.4 B-4

待翻译程序

```
int fib(int n) {
        if(n == 0)
                return 1;
        else if(n == 1)
                return 1;
        else
                return fib(n-1)+fib(n-2);
}

int main() {
        write(fib(5));
        write(fib(4));
        write(fib(3));
```

```
        write(fib(2));
        write(fib(1));
        write(fib(0));
        return 0;
}
```

程序输入：无；预期输出：8 5 3 2 1 1

说明：这个用例主要测试针对函数调用的翻译，程序使用递归的方式计算菲波纳契数列。

3 C 组测试用例

本组测试用例是一个相对比较复杂的程序，语义是计算一个范围内的水仙花数，并将这些数字以 16 进制的形式打印。由于这里的虚拟机只支持输出整数，所以如果要输出 A-F，就以他们对应的数字的相反数来代替（例如用 -15 代替 F）。

```
int power(int base1, int p1) {
        int ret1 = 1;
        while(p1>0)
        {
                ret1 = ret1 * base1;
                p1 = p1 - 1;
        }
        return ret1;
}

int mod(int number2, int m2)
{
        return number2 - number2 / m2 * m2;
}

int getNumDigits(int number3)
{
        int ret3 = 0;
        if(number3 < 0)
```

```
20                  return -1;
21          while(number3 > 0) {
22                  number3 = number3 / 10;
23                  ret3 = ret3+1;
24          }
25
26          return ret3;
27  }
28
29  int isNarcissistic(int number4)
30  {
31          int numDigits4 = getNumDigits(number4);
32          int sum4 = 0;
33          int n4 = number4;
34          int s4;
35          while(n4>0) {
36                  s4 = mod(n4, 10);
37                  n4 = (n4 - s4) / 10;
38                  sum4 = sum4+power(s4, numDigits4);
39          }
40
41          if(sum4 == number4)
42                  return 1;
43          else
44                  return 0;
45  }
46
47  int printHexDigit(int digit6)
48  {
49          if(digit6 < 10)
50                  write(digit6);
51          else
```

```
52                  write(-digit6);
53          return 0;
54  }
55
56  int printHex(int number5)
57  {
58          int a5[4];
59          int j5 = 0;
60          while(j5<4) {
61                  a5[j5] = mod(number5, 16);
62                  number5 = number5 / 16;
63                  j5=j5+1;
64          }
65          j5 = 3;
66          while(j5>=0) {
67                  printHexDigit(a5[j5]);
68                  j5=j5-1;
69          }
70          return 0;
71  }
72
73
74  int main() {
75          int count = 0;
76          int i = 9400;
77          while(i<9500)
78          {
79                  if(isNarcissistic(i) == 1)
80                  {
81                          printHex(i);
82                          count = count + 1;
83                  }
```

```
                i=i +1;
        }
        return count;
}
```

程序输入：无；预期输出：2 5 0 2

说明：这个用例中包含了所有的语言特性，9400-9500 中只有 9474 这一个符合要求的数，表示成 16 进制的形式是 2502。

4 D 组测试用例

这组测试用例主要用于测试中间代码的优化，程序中有多个可优化点，包括常量折叠，公共子表达式等。

```
int main()
{
        int a = 6+6-11*1, b = 7-4, c = 3+4+5*6/3-4;

        int d = a + b;
        int e = a + b + c;
        int f = a + b + c;
    int g1 = 42, i = 0;
    int g;

        f = a + b + c - d - e + f;

        while(f>b-a) {
                g = f - b + a*2 + c*d - f;
                g1 = g1 + i*4 + 3+4+5;
                i = i + 1 + 0;
                if(i-i/3*3 == a-a+b-b)
                        f = f + 2;
                f = f - 1;
        }
```

```
21          write(g1);
22
23          i = g1;
24          while(g1 >= 1200+22) {
25                  i = g1+1024;
26                  g1 = g1 -1;
27                  i = g1;
28          }
29          write(g1);
30          a = a + b;
31          b = a + b;
32          c = a + b;
33
34          write(c);
35          return 0;
36  }
```

程序输入：无；预期输出：2014 1221 11

说明：首先需要保证中间代码的正确性，要能准确输出最后的结果，才能参加后面的效率竞赛。30-32 行的这三行语句不能当作公共子表达式进行优化。

5 E 类测试用例

5.1 E3.1

本组测试用例主要针对 3.1 分组的同学，要求能翻译结构体。3.2 分组的同学只要提示无法进行翻译，并且不产生中间代码即可，如果产生了中间代码，将会被倒扣分。

待翻译程序 1：

```
1  struct Vector
2  {
3      int x, y;
4  };
5
6  int main()
```

```
{
    struct Vector v;
    v.x = 2;
    v.y = 4;
    write(v.x);
    return 0;
}
```

程序输入：无；预期输出：2

说明：这里使用了简单的结构体。

待翻译程序 2：

```
struct Vector
{
    int x, y;
};

int crossProduct(struct Vector v11, struct Vector v21)
{
    return v11.x*v21.y - v11.y*v21.x;
}

int main()
{
    struct Vector v1, v2;
    v1.x = 1;
    v1.y = 2;
    v2.x = 3;
    v2.y = 6;
    write(crossProduct(v1, v2));
    return 0;
}
```

程序输入：无；预期输出：0

说明：这里将结构体作为参数使用。

5.2 E3.2

本组测试用例主要针对 3.1 分组的同学，要求能翻译高维数组，并处理数组作为参数的情况。3.1 分组的同学只要提示无法进行翻译，并且不产生中间代码即可，如果产生了中间代码，将会被倒扣分。

待翻译程序 2：

```
int main()
{
    int a[2][2];
    a[0][0] = 0;
    a[0][1] = 2;
    a[1][0] = 2;
    a[1][1] = 0;
    write(a[0][0]+a[0][1] - a[1][0]-a[1][1]);
    return 0;
}
```

程序输入：无；预期输出：0

说明：这里将使用了高维数组。

待翻译程序 2：

```
int min(int a[10], int n)
{
        int i = 1, j = n;
    int min = 0;
        while(i<j) {
                if(a[i] < a[min]) {
                        min = i;
                }
                i = i+1;
        }

        return a[min];
```

```
13 }
14
15 int main()
16 {
17     int array[10];
18     int index = 0, size = 10;
19     while(index<size)
20     {
21         int ti = index*index*index;
22         array[index] = ti - ti/size*size;
23         index = index + 1;
24     }
25     write(min(array, size));
26 }
```

程序输入：无；预期输出：0

说明：这里将数组作为参数使用，求一个数组中的最小值。

6 结束语

如果对本测试用例有任何疑议，可以写邮件与奚旺助教联系，注意同时抄送给许老师。