

第4讲：分支和循环（下）

掌握了前面学习的这些知识，我们就可以写一些稍微有趣的代码了，比如：

写一个猜数字游戏

游戏要求：

1. 电脑自动生成1~100的随机数
2. 玩家猜数字，猜数字的过程中，根据猜测数据的大小给出大了或小了反馈，直到猜对，游戏结束

1. 随机数生成

要想完成猜数字游戏，首先得产生随机数，那怎么产生随机数呢？

1.1 rand

C语言提供了一个函数叫 `rand`，这函数是可以生成随机数的，函数原型如下所示：

```
1 int rand (void);
```

`rand`函数会返回一个伪随机数，这个随机数的范围是在0~`RAND_MAX`之间，这个`RAND_MAX`的大小是依赖编译器上实现的，但是大部分编译器上是32767。

`rand`函数的使用需要包含一个头文件是：`stdlib.h`

那我们就测试一下`rand`函数，这里多调用几次，产生5个随机数：

```
1 #include <stdio.h>
2 #include <stdlib.h>
3 int main()
4 {
5     printf("%d\n", rand());
6     printf("%d\n", rand());
7     printf("%d\n", rand());
8     printf("%d\n", rand());
9     printf("%d\n", rand());
10    return 0;
11 }
```

我们先运行一次，看看结果，再运行一次再看看结果，多运行几次呢？

```
C:\> 选择 Microsoft Visual Studio 调试控制台  
41  
18467  
6334  
26500  
19169
```

第一次运行结果

```
C:\> 选择 Microsoft Visual Studio 调试控制台  
41  
18467  
6334  
26500  
19169
```

第二次运行结果

我们可以看到虽然一次运行中产生的5个数字是相对随机的，但是下一次运行程序生成的结果和上一次一模一样，这就说明有点问题。

如果再深入了解一下，我们就不难发现，其实rand函数生成的随机数是伪随机的，伪随机数不是真正的随机数，是通过某种算法生成的随机数。真正的随机数的是无法预测下一个值是多少的。而rand函数是对一个叫“种子”的基准值进行运算生成的随机数。

之所以前面每次运行程序产生的随机数序列是一样的，那是因为rand函数生成随机数的默认种子是1。如果要生成不同的随机数，就要让种子是变化的。

1.2 srand

C语言中又提供了一个函数叫 `srand`，用来初始化随机数的生成器的，srand的原型如下：

```
1 void srand (unsigned int seed);
```

程序中在调用 rand 函数之前先调用 srand 函数，通过 srand 函数的参数seed来设置rand函数生成随机数的时候的种子，只要种子在变化，每次生成的随机数序列也就变化起来了。

那也就是说给srand的种子如果是随机的，rand就能生成随机数；在生成随机数的时候又需要一个随机数，这就矛盾了。

1.3 time

在程序中我们一般是使用程序运行的时间作为种子的，因为时间时刻在发生变化的。

在C语言中有一个函数叫 `time`，就可以获得这个时间，time函数原型如下：

```
1 time_t time (time_t* timer);
```

time 函数会返回当前的日历时间，其实返回的是1970年1月1日0时0分0秒到现在程序运行时间之间的差值，单位是秒。返回的类型是time_t类型的，time_t 类型本质上其实就是32位或者64位的整型类型。

time函数的参数 timer 如果是非NULL的指针的话, 函数也会将这个返回的差值放在timer指向的内存中带回去。

如果 timer 是NULL, 就只返回这个时间的差值。time函数返回的这个时间差也被叫做: 时间戳。

time函数的时候需要包含头文件: time.h

```
1 //VS2022 上time_t类型的说明
2
3 #ifndef _CRT_NO_TIME_T
4     #ifdef _USE_32BIT_TIME_T
5         typedef __time32_t time_t;
6     #else
7         typedef __time64_t time_t;
8     #endif
9 #endif
10
11
12 typedef long                __time32_t;
13 typedef __int64             __time64_t;
```

如果只是让time函数返回时间戳, 我们就可以这样写:

```
1 time(NULL); //调用time函数返回时间戳, 这里没有接收返回值
```

那我们就可以让生成随机数的代码改写成如下:

```
1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <time.h>
4
5 int main()
6 {
7     //使用time函数的返回值设置种子
8     //因为srand的参数是unsigned int类型, 我们将time函数的返回值强制类型转换
9     srand((unsigned int)time(NULL));
10    printf("%d\n", rand());
11    printf("%d\n", rand());
12    printf("%d\n", rand());
13    printf("%d\n", rand());
14    printf("%d\n", rand());
15    return 0;
16 }
```

多运行几次看看，每次的运行就有差异了。

Microsoft Visual Studio 调试控制台

```
28631
12047
22702
17901
8208
```

第一次运行的结果

Microsoft Visual Studio 调试控制台

```
28673
20705
25560
3044
11225
```

第二次运行的结果

(注：截图只是当时程序运行的结果，你的运行结果不一定和这个一样)

srand函数是不需要频繁调用的，一次运行的程序中调用一次就够了。

1.4 设置随机数的范围

如果我们要生成0~99之间的随机数，方法如下：

```
1 rand() %100; //余数的范围是0~99
```

如果要生成1~100之间的随机数，方法如下：

```
1 rand()%100+1; // %100的余数是0~99, 0~99的数字+1, 范围是1~100
```

如果要生成100~200的随机数，方法如下：

```
1 100 + rand()%(200-100+1)
2 //余数的范围是0~100, 加100后就是100~200
```

所以如果要生成a~b的随机数，方法如下：

```
1 a + rand()%(b-a+1)
```

2. 猜数字游戏实现

参考代码：

```
1  #include <stdio.h>
2  #include <stdlib.h>
3  #include <time.h>
4
5
6  void game()
7  {
8      int r = rand()%100+1;
9      int guess= 0;
10     while(1)
11     {
12         printf("请猜数字>:");
13         scanf("%d", &guess);
14         if(guess < r)
15         {
16             printf("猜小了\n");
17         }
18         else if(guess > r)
19         {
20             printf("猜大了\n");
21         }
22         else
23         {
24             printf("恭喜你, 猜对了\n");
25             break;
26         }
27     }
28 }
29
30 void menu()
31 {
32     printf("*****\n");
33     printf("*****  1. play  *****\n");
34     printf("*****  0. exit  *****\n");
35     printf("*****\n");
36 }
37
38 int main()
39 {
40     int input = 0;
41     srand((unsigned int)time(NULL));
```

```
42     do
43     {
44         menu();
45         printf("请选择:>");
46         scanf("%d", &input);
47         switch(input)
48         {
49             case 1:
50                 game();
51                 break;
52             case 0:
53                 printf("游戏结束\n");
54                 break;
55             default:
56                 printf("选择错误, 重新选择\n");
57                 break;
58         }
59     }while(input);
60     return 0;
61 }
```

还可以加上猜数字的次数限制, 如果5次猜不出来, 就算失败.

```
1  #include <stdio.h>
2  #include <stdlib.h>
3  #include <time.h>
4
5  void game()
6  {
7      int r = rand() % 100 + 1;
8      int guess = 0;
9      int count = 5;
10     while (count)
11     {
12         printf("\n你还有%d次机会\n", count);
13         printf("请猜数字>:");
14         scanf("%d", &guess);
15         if (guess < r)
16         {
17             printf("猜小了\n");
18         }
19         else if (guess > r)
20         {
21             printf("猜大了\n");
```

```
22     }
23     else
24     {
25         printf("恭喜你,猜对了\n");
26         break;
27     }
28     count--;
29 }
30 if (count == 0)
31 {
32     printf("你失败了,正确值是:%d\n", r);
33 }
34 }
35
36 void menu()
37 {
38     printf("*****\n");
39     printf("***** 1. play *****\n");
40     printf("***** 0. exit *****\n");
41     printf("*****\n");
42 }
43
44 int main()
45 {
46     int input = 0;
47     srand((unsigned int)time(NULL));
48     do
49     {
50         menu();
51         printf("请选择:>");
52         scanf("%d", &input);
53         switch (input)
54         {
55             case 1:
56                 game();
57                 break;
58             case 0:
59                 printf("游戏结束\n");
60                 break;
61             default:
62                 printf("选择错误,重新选择\n");
63                 break;
64         }
65     } while (input);
66     return 0;
67 }
```

完

比特就业课