跳转结构

2022 年 8 月 16 日

1. 输入一个班的成绩，求总分、平均分、最低分和最高分，成绩输入以-1 表示结束。

```c
#include <stdio.h>
int main()
{
    int sum_score = 0;
    int student_num = 0;
    int max_score = -1;
    int min_score = 101;
    while(1)
    {
        int score = 0;
        printf("Input a student score:\n");
        scanf("%d", &score);
        if(score == -1)
        {
            break;
        }
        sum_score += score;
        student_num++;
        if(max_score < score)
        {
            max_score = score;
        }
        if(min_score > score)
        {
            min_score = score;
        }
    }
```

```
    printf("sum_score: %d\n", sum_score);
    printf("average_score: %.2f\n", (float)sum_score / (float)student_num);
    printf("max_score: %d\n", max_score);
    printf("min_score: %d\n", min_score);
    return 0;
}
```

2. 给定一个正整数 n，判断它是否素数。

```
#include <stdio.h>
int main()
{
    int n;
    printf("Input a positive integer:\n");
    scanf("%d", &n);
    for(int i = 2; i < n; i++)
    {
        if(n % i == 0)
        {
            printf("no");
            return 0;
        }
    }
    printf("yes");
    return 0;
}
```

3. 以下实现一个简单的猜数字游戏。要求如下。

   (a) 所猜的数字为 0~9 之间的整数。

   (b) 请求用户做出猜测后，如果用户给出的猜测不在上述范围内，就重新请求，直到用户的猜测处在上述范围中。

   (c) 每一次用户做出上述范围中的猜测之后，如果用户猜中，就提示用户猜中；否则，就提示用户偏大还是偏小，并允许用户继续猜测。

   (d) 每一次用户猜中之后，要询问用户是否继续游戏。用户只允许输入 y 或者 n。如果用户的输入不符合要求，就重新询问用户，直到用户的输入符合要求。

(e) 如果用户输入 n，则终止游戏，退出程序。否则，重新生成一个在上述范围内的随机数，再次进行上述过程。

```c
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
int main()
{
   srand(time(NULL));
   // 以下有四个死循环形式，把跳出的任务交给break，使得编程更灵活
   do
   {
      int magic_num = rand() % 10; // 产生0~9之间的随机数
      int guess_num; // 用于存储用户猜测的数字
      while(1)
      {
         while(1)
         {
            printf("make a guess (0~9):\n");
            scanf("%d", &guess_num);
            if(guess_num >= 0 && guess_num <= 9) // 所猜数字处在合适的范围内
            {
               break; // 跳出猜测循环
            }
            // 否则就重猜
         }
         if(guess_num == magic_num) // 猜中
         {
            printf("correct\n");
            break; // 跳出猜测当前数字的循环
         }
         else if(guess_num > magic_num) // 偏大
         {
            printf("too large\n"); // 提示用户
         }
         else // 偏小
         {
            printf("too small\n");
         }
```

```
37        }
38        char ch; // 用于存储用户关于是否继续的回答
39        while(1)
40        {
41           getchar(); // 吸收之前多余的空格符
42           printf("would like to continue? (y/n)\n");
43           ch = getchar(); // 读取用户的输入
44           if(ch == 'y' || ch == 'n') // 用户的输入符合要求
45           {
46              break; // 跳出询问是否继续的环节
47           }
48           // 否则就继续询问用户
49        }
50        if(ch == 'n') // 用户不想继续
51        {
52           break; // 跳出游戏
53        }
54     } while(1);
55     return 0;
56  }
```

4. 题目同上。

```
1   #include <stdio.h>
2   #include <stdlib.h>
3   #include <time.h>
4   int main()
5   {
6      srand(time(NULL));
7      // 以下根据逻辑关系，并巧妙使用break和continue，使得编程更灵活
8      do
9      {
10        int magic_num = rand() % 10; // 产生0~9之间的随机数
11        int guess_num; // 用于存储用户猜测的数字
12
13        while(1)
14        {
15           printf("make a guess (0~9):\n");
16           scanf("%d", &guess_num);
```

```
17
18          if(guess_num < 0 || guess_num > 9) // 所猜数字不在合适的范围内
19          {
20              continue; // 重猜
21          }
22
23          if(guess_num > magic_num) // 偏大
24          {
25              printf("too large\n"); // 提示用户
26              continue; // 继续猜
27          }
28          else if(guess_num < magic_num) // 偏小
29          {
30              printf("too small\n");
31              continue; // 继续猜
32          }
33
34          // 这次的数字被猜中
35          printf("correct\n");
36          break; // 退出猜这个数字的循环
37      }
38
39      char ch; // 用于存储用户关于是否继续的回答
40      while(1)
41      {
42          getchar(); // 吸收之前多余的空格符
43          printf("would like to continue? (y/n)\n");
44          ch = getchar(); // 读取用户的输入
45          if(ch != 'y' && ch != 'n') // 用户的输入不合要求
46          {
47              continue; // 继续询问
48          }
49          // 用户的输入符合要求
50          break; // 退出询问是否继续的循环
51      }
52
53      if(ch == 'y') // 用户想继续
54      {
55          continue; // 继续玩
```

```
    }

    // 用户不想继续
    break;

  } while(1);

  return 0;
}
```

5. 已知用户依次输入 3 个或以上正整数（输入以-1 表示结束），要求编写程序，输出相差最大的相邻两个数（若存在多对整数满足要求，则只输出其中一对）。

```
#include <stdio.h>
#include <math.h>
int main()
{
  int former, latter; // 用于保存相邻的两个数，一前一后

  // 先存储最初输入的两个数
  printf("input two integers:\n");
  scanf("%d%d", &former, &latter);

  int max_diff = abs(former - latter); // 默认它们的差是最大的

  // 默认它们就是我们想要的答案
  int result1 = former;
  int result2 = latter;

  // 下面遍历用户的全部输入
  while(1)
  {
    // 更新former
    former = latter;
    printf("input an integer (-1 means the end):\n");
    // 更新latter
    scanf("%d", &latter);
    if(latter == -1) // 输入结束
    {
```

```
            break;
        }
        if(max_diff < abs(former - latter)) // 发现差值更大的
        {
            // 更新得到的解
            result1 = former;
            result2 = latter;
            // 更新差值
            max_diff = abs(former - latter);
        }
    }

    printf("A satisfying pair is (%d, %d).\n", result1, result2);

    return 0;
}
```

6. 如果正整数 a，b 和 c 满足 $a^2+b^2=c^2$，则称 a，b 和 c 为一组勾股数，并且称 c 为“勾股数的斜边数”。编写程序，请求用户输入正整数 n，并输出大于 n 的最小“勾股数的斜边数”。

```
#include <stdio.h>
int main()
{
    int n;
    printf("input a positive integer:\n");
    scanf("%d", &n);
    n++; // 因为要求找到的数比用户输入的要大，所以这里从下一个开始考虑
    while(1) // 因为循环继续进行的条件稍稍复杂，这里写死循环能简化代码
    {
        for(int x = 1; x < n; x++) // 勾股数满足直角边小于斜边
        {
            int y;
            // 以下无循环体，如果平方和超出，就不要再试
            for(y = 1; x * x + y * y <= n * n; y++);
            if(x * x + y * y == n * n) // 满足勾股数的要求
            {
                printf("having just found %d\n", n); // 因为从小到大逐个检查，这一定是最小的
                return 0; // 程序结束
```

```
19            }
20        }
21    }
22 }
```