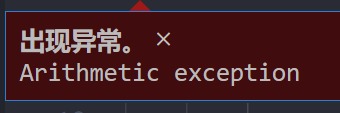
# 调试作业 8\_02

## 一、下面是一个判断素数的函数，通过调试找到错误并修改

#include <stdio.h>  
#include <math.h>  
#include <stdbool.h>  
   
bool IsPrimer(int n)  
{  
 for (int i = 0; i < sqrt(n \* 1.0); i++)  
 {  
 if (n % i == 0)  
 return false;  
 else  
 return true;  
 }  
 return false;  
}  
   
int main()  
{  
 for (int i = 0; i < 100; i++)  
 {  
 if (IsPrimer(i))  
 printf("%d 是素数\n", i);  
 }  
 return 0;  
}

### 调试之后出现错误如下



调试结果

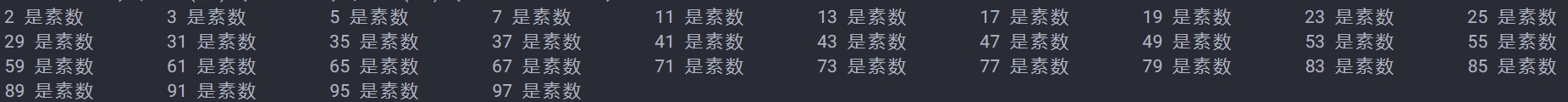
### 错误分析

1. 可确定是计算方法出现错误, 定位原因发现**源代码 12 行**初始化变量 i 时初始值为 0 , 导致第 14 行if (n % i == 0) 出现了计算错误.
2. 进一步分析, 在 isprime 函数中, i 的值不能初始化为 **1** ,该行为会导致无论传入的形参为何值, 都会返回 **false** 值.
3. 而在 for (int i = 0; i < sqrt(n \* 1.0); i++) 此行代码中, 使用<而非<=会导致部分非素数被输出; 单次加一的步长也会略微影响性能.
4. 缺少形参小于等于二时的判断, 会导致 2 作为素数无法输出.

### 修改后程序

#include <stdio.h>  
#include <math.h>  
#include <stdbool.h>  
   
bool IsPrimer(int n)  
{  
 if (n == 1 || (n % 2 == 0 && n != 2))  
 {  
 return false;  
 }  
 for (int i = 3; i <= sqrt(n) + 1; i += 2)  
 {  
 if (n % i == 0)  
 return false;  
 else  
 return true;  
 }  
 return false;  
}  
   
int main()  
{  
 for (int i = 0; i < 100; i++)  
 {  
 if (IsPrimer(i))  
 printf("%d 是素数\n", i);  
 }  
 return 0;  
}

### 运行结果

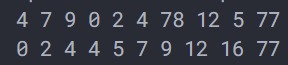


运行结果

## 二、下面是一个冒泡程序，通过调试找到错误并修改

#include <stdlib.h>  
#include <stdio.h>  
   
void BubbleSort(int arr[10], int len)  
{  
 int tmp;  
 for (int i = 0; i < len; i++)  
 {  
 for (int j = 0; j < len; j++)  
 {  
 if (arr[j] > arr[j + 1])  
 {  
 tmp = arr[j];  
 arr[j] = arr[j + 1];  
 arr[j + 1] = tmp;  
 }  
 }  
 }  
}  
   
void Show(int arr[10], int len)  
{  
 for (int i = 0; i < len; i++)  
 {  
 printf("%d ", arr[i]);  
 }  
 printf("\n");  
}  
   
int main()  
{  
 int arr[10] = {4, 7, 9, 0, 2, 4, 78, 12, 5, 77};  
 Show(arr, 10);  
 BubbleSort(arr, 10);  
 Show(arr, 10);  
   
 return 0;  
}

### 调试后错误如下



错误显示

### 错误分析

1. 可见, 输出的数组**缺失了 78**, **多余了 16**.
2. 分析, 发现代码行 for (int j = 0; j < len; j++) 在指针指向数组边界时, 会导致下方的交换部分**越界**, 导致数值错误.

### 修改后程序

#include <stdlib.h>  
#include <stdio.h>  
   
void BubbleSort(int arr[10], int len)  
{  
 int tmp;  
 for (int i = 0; i < len; i++)  
 {  
 //此处为修改.  
 for (int j = 0; j < len - i -1; j++)  
 {  
 if (arr[j] > arr[j + 1])  
 {  
 tmp = arr[j];  
 arr[j] = arr[j + 1];  
 arr[j + 1] = tmp;  
 }  
 }  
 }  
}  
   
void Show(int arr[10], int len)  
{  
 for (int i = 0; i < len; i++)  
 {  
 printf("%d ", arr[i]);  
 }  
 printf("\n");  
}  
   
int main()  
{  
 int arr[10] = {4, 7, 9, 0, 2, 4, 78, 12, 5, 77};  
 Show(arr, 10);  
 BubbleSort(arr, 10);  
 Show(arr, 10);  
 return 0;  
}

### 运行结果

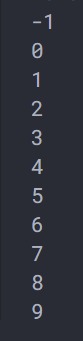
运行结果

运行结果

## 三、下面是一个二分查找的代码，通过调试找到错误并修改

#include <stdlib.h>  
#include <stdio.h>  
   
int SearchBin(int arr[10], int len, int key)  
{  
 int low = 0;  
 int high = len;  
 int mid;  
 while (low < high)  
 {  
 mid = (low + high) / 2;  
 if (arr[mid] == key)  
 return mid;  
 else if (arr[mid] < key)  
 low = mid;  
 else  
 high = mid;  
 }  
 return -1; //查找失败  
}  
   
int main()  
{  
 int arr[10] = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10};  
 for (int i = 0; i < 15; i++)  
 {  
 int index = SearchBin(arr, 10, i);  
 printf("%d\n", index); //查看下标是否正确  
 }  
 return 0;  
}

### 调试后出现错误如下

 之后程序进入死循环

### 错误分析

1. 程序查找进行到 10 之后, 继续进行查找会造成访问数组越界, 导致程序进入死循环.
2. 初步判断为在key值大于数组的最大数值时, 低位指针和中间指针会重合, 高位指针将不会发生改变, 导致程序陷入死循环.
3. 解决方案只需要在每次刷新高, 低, 中位指针时错位, 避免低位指针与中位指针重叠, 并且改变循环条件, 使得高低指针重叠时依旧可以进入循环.

*同时改变输出格式, 便于观察.*

### 修改后程序

#include <stdlib.h>  
#include <stdio.h>  
   
int SearchBin(int arr[10], int len, int key)  
{  
 int low = 0;  
 int high = len;  
 int mid;  
 while (low <= high)  
 {  
 mid = (low + high) / 2;  
 if (arr[mid] == key)  
 return mid;  
 else if (arr[mid] < key)  
 low = mid + 1;  
 else  
 high = mid - 1;  
 }  
 return -1; //查找失败  
}  
   
int main()  
{  
 int arr[10] = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10};  
 for (int i = 0; i < 15; i++)  
 {  
 int index = SearchBin(arr, 10, i);  
 printf("%d ", index); //查看下标是否正确  
 }  
 return 0;  
}

### 运行结果

运行结果

运行结果

#### 另附第四题

#include <stdlib.h>  
#include <stdio.h>  
   
int main()  
{  
 char arr[] = "abcdefghijk";  
 char brr[20];  
 int lenarr = sizeof(arr) / sizeof(arr[0]);  
 int lenbrr = sizeof(brr) / sizeof(brr[0]);  
   
 for (int i = 0; i < lenarr; i++)  
 {  
 brr[i] = arr[i];  
 }  
 for (int i = 0; i < lenbrr; i++)  
 {  
 printf("%c", brr[i]);  
 }  
   
   
 return 0;  
}