计算机学院　高级语言程序设计　课程实验报告

实验题目：异常处理		学号：202200400053
日期：2024-06-13	班级： 2202	姓名： 王宇涵
Email：1941497679@qq.com		

实验目的：

1. 掌握异常处理的执行过程和会使用异常处理。

实验软件和硬件环境：

软件环境 ：VSCODE + DEV-C++

硬件环境 ：Legion Y7000P

实验步骤与内容：

1. 仿照第 12 章 PPT，例 12_2。完成实验 12，lab12_1.cpp。自定义异常类

```
PS D:\BaiduSyncdisk\CLASSES\C++\exp\C++V4实验\ch12\lab12_1\output> & .\'lab12_1.exe'
CException
```

自定义异常类: 在 CException 类的成员函数 Reason()中用 cout 显示异常的类型,在函数 fn1()中用 throw 语句触发异常,在主函数的 try 模块中调用 fn1(),在 catch 模块中捕获异常。

2. 练习标准程序库中的异常类。第 12 章 PPT，例 12_3。

```
-1
std::bad_array_new_length
PS D:\BaiduSyncdisk\CLASSES\C++\exp\exp13\code\output>
```

输入-1, 报错数组长度有误

3. 练习习题 12-4。 异常类继承

```
Enter an integer(0~~1000): 1100
Number out of range. You used 1100 !
```

```
Enter an integer(0~~1000): 50
The number is : 50
```

数组越界: 直接返回报错信息

数组在范围内: 使用*p 进行数字的打印

4. 练习习题 12-5，处理内存分配异常。

书示例代码有误。需用 catch(const char * str 或 char const * str)匹配。

```
buf = new char[10000000000000000];
// 使用内存缓冲区 buf
```

```
PS D:\BaiduSyncdisk\CLASSES\C++\exp\exp13\code\output> & .\'ex12-5.exe'
有异常产生：内存分配失败！错误信息：std::bad_alloc
```

数组分配过大空间, 产生报错信息.

5. 练习习题 12-6。 Array 类模板，下标出界异常处理。

```cpp
Array<int> intArray(12);
//use try catch
try {
    intArray[20] = 7;
} catch (std::out_of_range& e) {
    std::cerr << "Caught an exception: " << e.what() << std::endl;
}
```

```
PS D:\BaiduSyncdisk\CLASSES\C++\exp\exp13\code\output> & .\'ex12-6.exe'
Caught an exception: invalid position
```

初始化数组长度为 12，输入 20 下标异常报错

6. 完成课本 12.5，综合实例，例 12-4。

```
(a)add account (d)deposit (w)withdraw (s)show (c)change day (n)next month (q)query (e)exit
2009-1-1        Total: 1802.36  command> a x invalidAccountType 0.05         // 非法的账户类型，应该是's'或'c'
2009-1-1        #invalidAccountType created
2009-1-1        Total: 1802.36  command> a s savings1 -0.01                 // 无效的利率，应该是非负数
2009-1-1        #savings1 created
2009-1-1        Total: 1802.36  command> d 0 1000                           // 尝试向不存在的账户存款
2009-1-1        #savings1       1000    2002.36                            // 尝试向不存在的账户存款
2009-1-1        Total: 2802.36  command> w 0 100                            // 尝试从不存在的账户取款
2009-1-1        #savings1       -100    1902.36                            // 尝试从不存在的账户取款
2009-1-1        Total: 2702.36  command> c 32                               // 无效的日期，超出了当月最大天数
Invalid day2009-1-1     Total: 2702.36  command> c -1                              // 无效的日期，负数日期
You cannot specify a previous day2009-1-1       Total: 2702.36  command> d 0 -500 deposit with negative     // 无效的存款金额，不能为负
2009-1-1        #savings1       -500    1402.36  deposit with negative     // 无效的存款金额，不能为负
2009-1-1        Total: 2202.36  command> w 0 -100 withdraw with negative    // 无效的取款金额，不能为负
2009-1-1        #savings1       100     1502.36  withdraw with negative    // 无效的取款金额，不能为负
2009-1-1        Total: 2302.36  command> n                                  // 在无账户时进入下一个月
2009-2-1        #credit1        -23.25  -73.25   interest
2009-2-1        Total: 2279.11  command> q 2008-13-01 2008-12-01            // 无效的日期，月份超出范围
2009-2-1        Total: 2279.11  command> q 2008-12-01 2008-11-01            // 无效的日期区间，结束日期早于开始日期
2009-2-1        Total: 2279.11  command> e                                  // 正常退出命令
PS D:\BaiduSyncdisk\CLASSES\C++\exp\exp13\code\output>
```

如图所示，输入了一些测试异常样例，产生了异常报错信息

结论分析与体会：

在完成本次实验的过程中，我深入学习了异常处理的执行过程，并掌握了如何在 C++程序中有效地使用异常处理机制。

通过实现自定义异常类，我学会了如何创建特定的异常类型，以便更好地描述和处理程序中的特定错误情况，这使得程序的可读性和可维护性大大增强。

在练习标准库中的异常类时，我理解了如何使用`std::exception`及其派生类来捕获和处理各种类型的标准异常。

此外，通过实现一个包含异常处理的`Array`类模板和处理内存分配异常，我进一步认识到异常处理在确保程序健壮性和可靠性方面的重要性。

在完成这些实验内容后，我对异常处理有了全面的理解，并能够在实际编程中灵活运用这些知识来编写健壮和高效的 C++程序。

就实验过程中遇到的问题及解决处理方法，自拟 1－3 道问答题：

1. 检测异常的作用?

答：异常检测在编程中的主要作用是确保程序的健壮性和可靠性。当程序运行过程中出现错误时，通过异常检测可以及时捕获这些错误，并采取适当的措施进行处理，避免程序崩溃。此外，通过使用异常处理机制，可以将错误处理代码与正常逻辑代码分离，使代码更加清晰易读。

2. 如何高效地检测异常?

答：尽量使用 C++标准库提供的异常类，如 std::exception 及其派生类。这些类经过优化，能够有效地捕获和处理异常。并且尽量避免捕获所有异常：不要使用 catch(...)来捕获所有异常，这样会使得异常处理不明确，难以确定具体的错误原因。