实验四：整数安全函数的设计

一、实验目标

1. 第二章-字符串、第五章-整数安全

2. 掌握常用的字符串函数；

3. 了解微软的 SafeInt 库（https://msdn.microsoft.com/en-us/library/dd570023.aspx）；

4. 理解整数溢出问题。

二、内容与要求

1. 所在模块：MathSafe

2. 编写函数：**int8_t my_atoi8(char* str, int* of);**

功能：将 str 指向的整数字符串转换为一个整数类型。本函数只针对[-128,127]范围内的整数执行转换。

参数 str：C 字符串。

参数 of：整数指针，返回转换中的错误情况。Str 合规且转换成功，(*of) = 0；str 不指向数字或者为空，(*of) = -1；str 指向的数字不合法，(*of) = -2.

返回值：返回转换后的整数，如果参数 str 不正确或者指向的数字越界，则返回 0.

提示：p164 例程改造。

3. 编写有符号整数运算函数：

（1）int iAdd_Safe(int8_t a, int8_t b, int* of);　功能：a+b

（2）int iSubtract_Safe(int8_t a, int8_t b, int* of); 功能：a-b

（3）int iMultiply_Safe(int8_t a, int8_t b, int* of); 功能：a*b

（4）int iDivide_Safe(int8_t a, int8_t b, int* of); 功能：a/b

功能：分别实现有符号数 int8_t 的加减乘除运算。

参数 of：整数指针，返回计算中的错误情况。(*of) = 0 计算成功；(*of) = -1 表示除零错；(*of) = -2 表示溢出错。

返回值：计算成功返回计算结果，否则返回 0.

4. 编写无符号整数运算函数：

（1）UINT Add_Safe(uint8_t a, uint8_t b, int* of);　功能：a+b

（2）UINT Subtract_Safe(uint8_t a, uint8_t b, int* of); 功能：a-b

（3）UINT Multiply_Safe(uint8_t a, uint8_t b, int* of); 功能：a*b

（4）UINT Divide_Safe(uint8_t a, uint8_t b, int* of)；功能：a/b

功能：分别实现无符号数 uint8_t 的加减乘除运算。

参数 of：整数指针，返回计算中的错误情况。(*of) = 0 计算成功；(*of) = -1 表示除零错；(*of) = -2 表示回绕错。

返回值：计算成功返回计算结果，否则返回 0.

5. 在 main 函数中对所编写函数进行测试，说明所使用的测试用例。

6. MSDN: https://msdn.microsoft.com/en-us/library/hh875057.aspx

7. MSDN: SafeInt Library, https://msdn.microsoft.com/en-us/library/dd570023.aspx

8. 建议学时：2 学时。

三、参考测试用例

序号	**int8_t my_atoi8(char* str, int* of);**	
1	**my_atoi8(“-128”), my_atoi8(“127”), my_atoi8(“0”), my_atoi8(“-1”)**	(*of) = 0
2	**my_atoi8(“1285”), my_atoi8(“-129”), my_atoi8(“130”)**	(*of) = -2
3	**my_atoi8(“”), my_atoi8(0), my_atoi8(“abc”), my_atoi8(“12a”)**	(*of) = -1
4	**my_atoi8(“123a”), my_atoi8(“12a”), my_atoi8(“-12a”)**	(*of) = -1
5	**my_atoi8(“00123”), my_atoi8(“-00123”)**	(*of) = -2
6	**my_atoi8(“ 123 ”), my_atoi8(“-23 ”), my_atoi8(“-2 3 ”)**	(*of) = 0

序号	**有符号整数加减乘除**	
1	**iAdd_Safe(12,13), iAdd_Safe(127,0),** **iAdd_Safe(-12, -13), iAdd_Safe(-127, 13)**	(*of) = 0
2	**iAdd_Safe(127,3), iAdd_Safe(-126,-13),**	(*of) = -2
3	**iSubtract_Safe(12,13), iSubtract_Safe(127,0),** **iSubtract_Safe(-12, -13), iSubtract_Safe(-127, 1)**	(*of) = 0
4	**iSubtract_Safe(127,-1), iSubtract_Safe(-126,13),**	(*of) = -2
5	**iMultiply_Safe(2,30), iMultiply_Safe(127,1), iMultiply_Safe(-2,30),** **iMultiply_Safe(-12, -3), iMultiply_Safe(-128, 1)**	(*of) = 0

6	**iMultiply_Safe(127,3), iMultiply_Safe(-28,6), iMultiply_Safe(-128,-1),**	(*of) = -2
7	**iDivide_Safe(127,3), iDivide_Safe(-28,7), iDivide_Safe(-128,-1),**	(*of) = 0
8	**iDivide_Safe(-128,-1),**	(*of) = -2
9	**iDivide_Safe(127,0), iDivide_Safe(-28,0), iDivide_Safe(-128,0),**	(*of) = -1

序号	**无符号整数加减乘除**	
1	**Add_Safe(12,13), Add_Safe(255,0), Add_Safe(132, 13)**	(*of) = 0
2	**Add_Safe(255,1), Add_Safe(124,123)**	(*of) = -2
3	**Subtract_Safe(17,13), Subtract_Safe(127,0), Subtract_Safe(232, 13)**	(*of) = 0
4	**Subtract_Safe(127,128), Subtract_Safe(2,255)**	(*of) = -2
5	**Multiply_Safe(2,30), Multiply_Safe(127,1), Multiply_Safe(122,2),**	(*of) = 0
6	**Multiply_Safe(127,3), Multiply_Safe(28,26)**	(*of) = -2
7	**Divide_Safe(234,3), Divide_Safe(228,7)**	(*of) = 0
8	**iDivide_Safe(127,0), iDivide_Safe(0,0)**	(*of) = -1

实验五：SafeInt 库的使用

一、实验目标

1. 了解整数运算过程可能出现的错误
2. 了解微软的整数安全数据类型—SafeInt 类
3. 了解微软的整数安全运算函数—SafeInt 函数
4. 理解 c++异常处理机制

二、内容与要求

1. 学会使用 SafeInt 类来完成整数安全运算。

   使用 SafeInt 类完成整数算术运算、逻辑运算与比较运算。

   使用 try、catch 块捕获算术溢出和除零错。

   头文件:<safeint.h>

   命名空间:msl:utilities

   SafeInt 类原型：

   ```
   template<typename T, typename E = _SAFEINT_DEFAULT_ERROR_POLICY>
   class SafeInt;
   ```

   创建 SafeInt 对象(示例):

   ```
   SafeInt<int> i1,i2(-5); SafeInt<uint8_t> u1,u2(10);
   ```

   执行加法运算(示例)：

   ```
   i1 = i2 + (uint8_t)u2;
   ```

2. 学会使用 SafeInt 函数来完成整数安全运算。

   使用 SafeInt 函数完成整数算术运算、逻辑运算与比较运算。

   示例：

   ```
   int8_t i1 = 126,res;int16_t i2 = 1,i3 = 2; bool b1,b2;
   b1 = SafeAdd(i1, i2, res);b2 = SafeAdd(i1, i3, res);
   ```

3. 掌握 c++异常处理机制

   在创建 SafeInt 对象时使用自定义异常处理策略。

   class MySafeIntException : public SafeIntException；

   要求：发生算术溢出时输出”SafeInt Arithmetic Overflow!”

   发生除零错误时输出”SafeInt Divide By Zero!”

三、参考测试用例

序号	**uint64_t+uint64_t**	
1	**0x000000007ffffffe, 0x0000000000000000**	**true**
2	**0xffffffffffffffff, 0x0000000000000001**	false
3	**0x000000007ffffffe, 0x000000007fffffff**	true
4	**0x7fffffffffffffff, 0x8000000000000001**	false

序号	**uint64_t+uint16_t**	
1	**0x7fffffffffffffff, 0x7ffe**	**true**
2	**0x8000000000000001, 0x8001**	true
3	**0xfffffffffffffffe, 0x8001**	false
4	**0xfffffffffffffffe, 0xffff**	false

序号	**uint64_t+int32_t**	
1	**0x000000007ffffffe, 0x7ffffffe**	**true**
2	**0xfffffffffffffffe, 0x7ffffffe**	false
3	**0x0000000000000001, 0xffffffff**	true
4	**0x0000000000000000, 0xffffffff**	false

序号	**int64_t+uint32_t**	
1	**0x000000007ffffffe, 0x00000001**	**true**
2	**0x7fffffffffffffff, 0x00000001**	true
3	**0x7fffffffffffffff, 0x00000001**	false
4	**0x8000000000000000, 0x00000001**	true

序号	**uint64_t-int32_t**	
1	**0x0000000000000000, 0x00000001**	**false**
2	**0x0000000000000000, 0x80000000**	true
3	**0xfffffffffffffffe, 0x80000000**	false

4	**0xfffffffffffffffe, 0xffffffff**	true

序号	**uint32_t*int64_t**	
1	**0x7fffffff, 2**	**true**
2	**0x80000000, 2**	false
3	**0x7fffffff, 0x7fffffff**	false
4	**1, 0x7fffffffffffffff**	false

序号	**uint64_t/int32_t**	
1	**0x7fffffff,0**	**false**
2	**0x80000000,0x7fffffff**	true
3	**0x80000000,0x80000000**	false
4	**1, 0xffffffff**	false

更多测试用例见：https://github.com/dcleblanc/SafeInt