西安电子科技大学
XIDIAN UNIVERSITY

面向对象程序设计

Object Oriented Programming

实验报告

Experimental Report

学号	20009200713	**姓名**	曾凡浩
班级	**2003052**	**任课教师**	**张淑平**
实验名称	第 1 次实验		
实验学期	**2021 – 2022 学年第 2 学期**		
实验日期	2022 年 3 月 31 日	**实验地点**	
报告成绩			

西安电子科技大学 计算机科学与技术学院

1、 实验目的

熟悉 C++的数据类型、表达式、基本的控制结构，能够灵活运用相应机制，提高编程能力。

2、 实验环境

操作系统：Window 10

开发工具：Visual Studio2020

3、 实验内容

3.1 题目 1 名称 ： 计算名字的相关数据

Read a sequence of possibly whitespace-separated (name,value) pairs, where the name is a single whitespace-separated word and the value is an integer or a floating-point value. Compute and print the sum and mean for each name and the sum and mean for all names. *Hint: §6.1.8*.

3.2 题目 2 名称 ： 定义函数 atoi()

Write a function *atoi*(*const char**) that takes a string containing digits and returns the corresponding *int*. For example, *atoi*("*123*") is *123*. Modify *atoi*() to handle C++ octal and hexadecimal notation in addition to plain decimal numbers. Modify *atoi*() to handle the C++ character constant notation.

3.3 题目 3 名称 ： 定义函数 itoa()

Write a function *itoa*(*int i*, *char b*[]) that creates a string representation of *i* in *b* and returns *b*.

4、 数据结构与算法说明

4.1 计算名字的相关数据

模块结构及文件组织设计：

模块 1：主控模块，仅包括文件 main.cpp ， 定义了 main()函数。

关键数据结构设计：

数据结构 1：定义了一个名字重要信息的结构体

```
struct information {
    int nameNum;    //记录该名字出现的次数;
```

```
        double sum;       //记录该名字 value 的和;
        information() {   //定义一个构造函数将上述信息初始化为 0;
            nameNum = 0;
            sum = 0;
        }
    };
    typedef map<string, information> myMap; //用一个 map 来记录信息
```

算法 1.1　int main()

作　用：主控函数，也实现对题目所需其他内容的测试。

参　数：无参数。

返回值：总是返回 0。

计算过程：

(1)　循环读入所有字符串。对于每个字符串看 map 中是否其出现过；

(2)　对于出现过的字符串在 map 中进行操作；

(3)　对于没出现过的字符串将其添加进 map；

(4)　用迭代器遍历 map。

4.2 定义函数 atoi()

模块结构及文件组织设计：

模块 1：主控模块，仅包括文件 main.cpp ，定义了 main()函数。

算法 2.1　int main()

作　用：主控函数，也实现对题目所需其他内容的测试。

参　数：无参数。

返回值：总是返回 0。

计算过程：

(1)　首先定义一个字符串常量；

(2)　通过其首部判断该字符串常量是什么进制表示；

(3)　将字符串赋值给对应表示进制的函数；

(4)　将其转化为数字后打印输出；

算法 2.2 `int atoi_10(const char* s)`

作　用：　将字符串转化为十进制数字；

参　数：　一个字符串常量；

返回值：　返回字符串常量转化后的十进制数字；

计算过程：

（1）先对字符串的符号进行判断，确定转化后数字的正负性；

（2）通过每个字符所处的位置给其赋权值；

（3）在转化的过程中判断转化后的字符是否超过 int 的范围，一旦超过返回 int 边界值退出函数；

算法 2.3`int atoi_8(const char* s)`

算法 2.4`int atoi_16(const char* s)`

与算法 2.2 类似，只是在转化过程中对不同数制直接的操作有一些细微差别，但是大体相同。

4.3 定义函数 itoa()（该程序可以完成整数的二进制，十进制，八进制和十六进制表示，但八进制和十六进制负数的原码表示）

模块结构及文件组织设计：

模块 1：主控模块，仅包括文件 main.cpp ，定义了 main()函数。

关键数据结构设计：

`int flag` //flag==0 表示正数，flag==1 表示负数，在输入数字后判断可将数字去除符号，方便后面对其操作。

算法 3.1　int main()

作　用：　主控函数，也实现对题目所需其他内容的测试。

参　数：　无参数。

返回值：　总是返回 0。

计算过程：

（1）定义一个足够的字符数组来保存转化后的数字字符；

（2）对于转化为二进制和八进制的数字先将其从十进制转化为对应进制；

（3）判断转化数字的位数，方便在写入字符数组时确定其下标；

（4）将转化好的字符数组遍历输出；

算法 3.2　`int change_2(int num);`

作　用：　将传入的十进制数字转化为二进制数字

参　数：　整型

返回值： 转化好的二进制数字

计算过程：循环取余直到原数为 0 时转化完成；

算法 3.3 `int change_8(int num;`

同上，区别仅为将十进制转化为八进制；

算法 3.4 `int digitNum(int num);`

作用：判断传入数字的位数；

参数：整型

返回值：数字位数

计算过程：循环除法求位数；

算法 3.5 `itoa(int num, char* buffer, int digit,int n)`

参数： 第一个为所需转化的数字，第二个是容纳转化后字符的字符数组，第三个是转化数字的位数，第四个为数字需转化为多少进制的数字字符；

返回值：无返回值

计算过程：

（1）先用 if 来判断需要转化为多少进制的数字字符；

（2）对 flag 进行判断，确定原数的正负，对字符数组首元进行赋值；

（3）对数字部分对其循环取余后与 ASCII 码表对应确定数字对应的数字字符；

5、 测试用例与测试结果

5.1 计算名字的相关数据

序号	测试数据	输出结果
1	xi 60 wei 50.2 jin 100 wei 40.5 wei 1.1 jin 200.0	name: jin number:2 average:150 name: wei number:3 average:30.6 name: xi number:1 average:60 totalSum:451.8 totalAverage:75.3
2	小王 20 小李 27.5 小曾 88.8 小王 10.5 小李 14 小张 40.2	name: 小李 number:2 average:20.75 name: 小王 number:2 average:15.25 name: 小曾 number:1 average:88.8 name: 小张 number:1 average:40.2 totalSum:201 totalAverage:33.5

5.2 定义函数 atoi()

序号	测试数据	转化结果
1	"123"	123
2	"-123"	-123
3	"0123"	83
4	"-0123"	-83
5	" \t+123."	123
6	"0xFG"	15
7	"-0789"	-7
8	"0x"	0
9	""	0
10	" "	0
11	"0x80000000"	-2147483648
12	"0x7FFFFFFF"	2147483647
13	"0xFFFFFFFF"	-1

5.3 定义函数 itoa()

序号	整数	10 进制串	16 进制串	8 进制串	2 进制串
1	123	123	0xB	0123	01111011
2	-123	-123	-0xB	-0123	11111011
3	83	83	0x3	083	01010011
4	-83	-83	-0x3	-083	11010011
5	303	303	0x2F	0303	0100101111
6	-303	-303	-0x2F	-0303	1100101111
7	2147483647	2147483647	0X7FFFFFFF	017777777777	溢出
8	-2147483647	-2147483647	0X80000001	020000000001	溢出
9	-2147483648	-2147483648	0X80000000	020000000000	溢出

6、 实验总结

在实验过程中对于 map 的运用不是十分熟练，通过实验分析学习到了对 map 的一些基本操作，对于字符串的转化问题常常做的是将数字字符转化为数字，但是本次实验中逆向转化的时候最开始无从下手，但其实两种转化是一样的，仅仅只需要在写入字符数组的时候对照 ASCII 码表即可，对于 8 进制和 16 进制负数的补码表示比较陌生所以不知道如何下手，课下会对这个部分重新学习。