



|  |  |  |  |
| --- | --- | --- | --- |
| **学号** | 18040400054 | **姓名** | **周师扬** |
| **班级** | **1803054** | **任课教师** | **张淑平** |
| **实验名称** | 简单类型、表达式和基本控制结构 | | |
| **实验学期** | **2019 – 2020 学年第2学期** | | |
| **实验日期** | 选择日期 | **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

# 实验目的

[熟悉 C++中的简单类型、基本表达式、基本的控制结构，能够灵活运用相应机制，提高编程能力。]

# 实验环境

操作系统：[MacOS]

开发工具：[Xcode]

# 实验内容

## **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.

## **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.

## **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.

# 数据结构与算法设计

## **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，Info\_message()函数。

**关键数据结构设计：**

数据结构1：定义了一个有关于name的结构体，用来存储一个name出现的次数和对应value的总和。

struct Name\_message//定义一个有关name信息的结构体

{

int count;//name输入的次数

double sum;//name对应的value的总和

Name\_message()//利用构造函数进行初始化新的Name\_message变量

{

count=0;

sum=0;

}

};

数据结构2:定义了一个map类型的变量dict\_type，用于存储键值对。

typedef map<string,Name\_message> dict\_type;//定义一个map类型

**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. 调用算法1.2，输入name值与value值；
2. 利用迭代器输出每个name值和对应的平均值和总和；
3. 输出总计的平均值和总和；

**算法1.2 void Info\_message()**

作 用：输入map模型的“键值对”。

参 数：无参数。

返回值：无返回值。

计算过程：

1. 当不挂起程序的时候持续输入“键值对”；
2. 判断name值是不是第一次出现；
3. 如果是，那么令count的值为1，sum的值为value；
4. 如果不是，那么count的值自增1，sum的值自增value；

## **定义函数atoi()**

**模块结构及文件组织设计：**

模块1：主控模块：仅包含文件main.cpp，定义了main()函数，用于测试所编写的头文件。

模块2：头文件模块：包含ATOI.hpp和ATOI.cpp，

ATOI.hpp头文件中包括：

Atoi()函数：将字符串转换为数字

Trans()函数：将字符转换成数字

qTrans()函数：将数字转换成字符

Decimal()函数：将字符串利用十进制转换成整数

Octonary()函数：将字符串利用八进制转换成整数

Hexadecima()函数：将字符串利用十六进制转换成整数

Scan()函数：扫描字符串并将转义写法变成常规字符串写法

ATOI.cpp链接文件用来解释ATOI.hpp文件给出的函数接口

**关键数据结构设计：**

数据结构1:string s，用于存储输入的字符串。

数据结构2:string p，用于存储经过Scan()函数扫描过的字符串。

数据结构3:char \* ptr=&s[0],定义一个指针用于移动和传输地址，既不用再次存储，又可以提高效率。

数据结构4:string c，用于存储等待转义的字符串。

**算法2.1 int main()**

作 用： 测试编写的头文件

参 数： 无参数

返回值： 总是返回0

计算过程：

1. 输入一个字符串s，并且利用Scan()函数进行扫描，转存为字符串p。
2. 利用eatspace()函数去除字符串首的空白。
3. 利用Atoi()函数进行转换并且输出转换后的change值进行测试。

**算法2.2 int Atoi()**

作 用： 将字符串转换为数字

参 数： **const** **char** \* s

返回值： 无返回值

计算过程：

1. 定义一个flag并且初始化为1，用于判断正负。
2. 定义一个char类型的指针ptr并指向字符串s的第一个字符，用于移动。
3. 判断第一位是否为’-’，如果是，那么令flag=-1。
4. 判断下一位是否为’0’，如果是，那么将ptr移动到下一位进行判断，如果为’x’或’X’，那么使用Hexadecima()函数进行转换，如果不是，那么使用Octonary()函数进行转换。
5. 如果下一位不为’0’，那么使用Decimal()函数进行转换。
6. 如果change超出了范围，那么返回change值为-1 。
7. 最后将change自乘flag值作为最终的转换值change进行返回。

**算法2.3 void Decimal();void Octonary();void Hexadecima()**

作 用： 将字符串利用十进制/八进制/十六进制转换成整数

参 数： const char \* ptr，int &change

返回值： 无返回值

计算过程：

1. 将ptr对应字符利用trans()转换为数字n。
2. 当ptr的下一位不为’0’时，将ptr右移一位并且将对应的字符利用trans()转换成数字m，并且循环使用公式n=n\*10/8/16+m得到最终的n值。
3. 将n值赋给change作为转换后的最终值。

**算法2.4 void Scan()**

作 用： 扫描字符串并将转义写法变成常规字符串写法

参 数： string a,string &b

返回值： 无返回值

计算过程：

1. 定义一个常量i标记字符串a的第一个字符，当i的下一位不为’\’时，将字符串a的值不断拷贝到b上。
2. 当i的下一位为’\’时，将i自加2，判断i的下一位是’0’还是’x’。如果是’0’，那么将之后两位存储到字符串c中并且利用Octonary()函数进行转换，然后拷贝到字符串b中。如果是’x’，那么利用Hexadecima()函数进行转换，然后拷贝到字符串b中。
3. 当i大于等于a的长度时，停止扫描，b即为扫描后的新的字符串。

## **定义函数itoa()**

**模块结构及文件组织设计：**

模块1:主控模块：仅包含文件main.cpp

定义了如下三个函数：

int main():用于测试两个自己编写的函数。

char\* Itoa():按进制将数字转换成字符串，8进制用‘0’作前缀，16进制用‘ 0x’作前缀。

void Trans():将字符转换成数字。

**关键数据结构设计：**

数据结构1:stack<char> s;

用于存储各位数据，并且利用push()函数、top()函数和pop()函数进行实现。

**算法3.1 int main()**

作 用： 测试Itoa()函数

参 数： 无参数

返回值： 始终返回0；

计算过程：

1. 定义一个待转换的整形数字integer，定义一个转换后的数组a[100]，定义进制表示整数base。
2. 使用Itoa()函数进行转换，并且使用测试样例进行测试。

**算法3.2 char\* Itoa()**

作 用： 按进制将数字转换成字符串

参 数： **int** I , **char** b[] , **int** base

返回值： 返回&b[0]，数组b的第一个元素的地址

计算过程：

1. 读取整数并且判断正负，如果是正数或0就令flag=1，如果是负数就令flag=-1。
2. 判断i是否是边界值，如果是，那么直接令对应的标记ts=1，并且将i值直接令为2147483647。
3. 使用abs()函数对整数i进行无符号化处理。
4. 将整数i转换成对应进制的数，并将每一位数据以字符的形式压入栈中。
5. 将8进制和16进制的对应前缀‘0’和‘0x’压入栈中。
6. 将flag值对应的正负号压入栈中。
7. 将栈中的数值反向填入数组中。
8. 如果边界值标记被点亮，那么令最后一位自加1。
9. 返回&b[0]，即数组b的第一个元素的首地址。

# 测试用例与测试结果

## **计算名字的相关数据**

**测试用例：**

zsy 200 wxl 300

zsy 300 wxl 600

wxl 100

**输出结果：**

按名字小计:

wxl 总和:1000 平均数:333.333

zsy 总和:500 平均数:250

总计:

总和:1500 平均数:300

## **定义atoi()函数**

**测试用例：**

123 -123 0123 -0123 01\\062\\063 1\\062\\x33 0x 空格 0x123 +0x123 0xF\\x66A 0xFFFFFFFF

**输出结果：**

1. -123 83 -83 83 123 0 0 291 291 4090 -1

## **定义itoa()函数**

**测试用例：**

数字与对应的进制表示：

123 10 8 16

233 10 8 16

567 10 8 16

-123 10

-244 10

**输出结果：**

123 123 0173 0x7B

233 233 0351 0XE9

567 567 01067 0X237

-123 -123

-244 -244

# 实验总结

本次实验一共包括三个上机题，第一道题主要是对之前作业知识的巩固，使用了map类型和之前学过的struct结构体，难度不算太大。第二题是模拟atoi()函数，让我们将进制转化的问题带到编程之中，进一步巩固了知识也进一步完善了编程的思维结构，还有一些重要的边界转换范围问题，还有空串等问题，让我更加注意到了这些细节，以后更不容易犯错。第三题其实是第二题的逆运算，已经实现了大部分要求，但是暂时没有完成负整数的8进制和16进制的准换，因为暂时不太明白补码的问题，期待以后能力更上一层楼的时候轻松解决这个问题。