



|  |  |  |  |
| --- | --- | --- | --- |
| **学号** | 18040400054 | **姓名** | **周师扬** |
| **班级** | **1803054** | **任课教师** | **张淑平** |
| **实验名称** | 函数编程 | | |
| **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++提供的函数相关机制，加深对命令行参数、参数传递方式、函数形参的特殊形式等知识的理解。掌握过程式程序设计范型的基本策略，能够灵活运用相应机制，提高程序设计能力]

# 实验环境

操作系统：[MacOS]

开发工具：[Xcode]

# 实验内容

## **命令行参数（选做）**

Write a program like ‘‘Hello, world!’’ that takes a name as a command-line argument and writes ‘‘Hello, name!’’. Modify this program to take any number of names as arguments and to say hello to each.

## **二叉树及其操作**

Consider:

struct Tnode{

string word;

int count;

Tnode \*left, \*right;

}

Write a function for entering new words into a tree of Tnodes.

Write a function to write out a tree of Tnodes.

Write a function to write out a tree of Tnodes with the words in alphabetical order.

## **可变参数表**

Write an error function that takes a printf-style format string containing %s, %c, and %d directives and an arbitrary number of arguments. Don’t use printf(). Look at §21.8 if you don’t know the meaning of %s, %c, and %d. Use <cstdarg>.

## **日期计算**

Write functions to add one day, one month, and one year to a Date as defined in §5.9[13].  
Write a function that gives the day of the week for a given Date.

Write a function that gives the Date of the first Monday following a given Date.

# 数据结构与算法设计

## **命令行参数**

**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：无关键数据结构

**算法1.1 int main()**

作 用：主控函数，输出命令行参数提供的值。

参 数：int argc, const char \* argv[]。

返回值：总是返回0。

计算过程：

使用循环和cout直接输出命令行参数提供的值。

## **二叉树及其操作**

**模块结构及文件组织设计：**

模块1：主控模块：包含文件main.cpp

定义了：

main()函数:用于测试所编写的头文件。

Search()函数:用于寻找节点，找到则返回true，否则返回false并且带回 节点位置

InOderTraverse()函数;中序遍历二叉树

Buynode()函数:生成新节点并且初始化

Insert()函数:向二叉树中插入节点

Destroy()函数:销毁二叉树

**关键数据结构设计：**

数据结构1:

**typedef** **struct** Tnode

{

string word;

**int** count;

Tnode \*left;

Tnode \*right;

}BSTNode, \*BSTree;

用于存储树的新节点

**算法2.1 int main()**

作 用： 测试编写的函数

参 数： 无参数

返回值： 总是返回0

计算过程：

1. 持续输入并且插入新节点
2. 中序遍历二叉排序树
3. 销毁二叉树

**算法2.2 void InOderTraverse()**

作 用： 中序遍历二叉树

参 数： **BSTree bst**

返回值： 无返回值

计算过程：

1. 如果节点有左孩子，那么递归实现对左孩子的操作。
2. 输出对应节点的值。
3. 如果有右孩子，那么递归实现对右孩子的操作。

**算法2.3 bool Insert()**

作 用： 向二叉排序树中插入新的节点

参 数： BSTree \*bst, string key

返回值： bool类型值true/false

计算过程：

1. 如果树为空树，那么直接令插入的节点为根结点并且返回true值。
2. 查找想要插入的节点的值。
3. 如果找到，则令该节点的count值自加1，并且停止插入返回false值。
4. 如果没找到，则记住Search()函数返回的最终指针的位置，生成新的节点并且初始化，插入到正确的位置(左孩子值小于key，右孩子值大于key)。

**算法2.4 bool Search()**

作 用： 在二叉排序树中找到key值对应的节点

参 数： BSTree bst, string key, BSTree f, BSTree \*p

返回值： bool类型值true/false

计算过程：

1. 定义一个可修改的指针p用于寻找制定的key值。
2. 如果该树为空树，那么令p=NULL，并且返回值false。
3. 如果key指小于p指向的节点的data值，那么查找p指向节点的左孩子，否则查找右孩子，并且递归实现。

**算法2.5 void destroy()**

作 用： 销毁使用c++New命令生成的二叉树

参 数： BSTree T

返回值： 无返回值

计算过程：

1. 如果该树已经为空树，则直接返回true。
2. 断开跟节点和他们左右孩子的联系，然后delete掉根结点并且递归实现。

## **可变参数表**

**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

定义了如下两个函数：

int main():用于测试两个自己编写的函数。

void error():用于模拟printf可变参数函数。

**关键数据结构设计：**

数据结构1:调用了<cstdarg>模块函数，并且使用了其中的va\_list、va\_start和va\_arg三个宏来实现对于可变参数的接收。

**算法3.1 int main()**

作 用： 测试error()函数

参 数： 无参数

返回值： 始终返回0；

计算过程：

1. 使用error()函数进行printf()函数的模拟，接收任意个数的字符并且输出，并且使用printf()函数来验证error()函数的正确性。

**算法3.2 void error()**

作 用： 模拟printf()函数

参 数： **const** **char** \*instruction,...（可变参数）

返回值：始终返回0

计算过程：

1. 定义一个va\_list的指针ap用来接收可变参数。
2. 使用va\_start接收其实固定参数。
3. 对接收的命令进行格式分析。
4. 使用va\_arg(ap,..)进行可变参数的接收。
5. 输出->模拟printf()函数。

## **日期计算**

**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

定义了如下若干函数：

int main():用于测试自己定义的函数。

bool If\_Leap():用于判断是否是闰年。

void Get\_Date():输入日期。

void Put\_Date():输出日期。

void Oprition():对日期的操作,’y’代表对年的操作，’m’代表对月的操作，’d’代表对天的操作。

void Op\_month():对月份的操作。

void Op\_day():对日的操作。

void F\_week():判断是星期几。

int F\_month():判断month的类型，使用flag值标记。

**关键数据结构设计：**

数据结构1:

**typedef** **struct** Date//定义一个Date类型

{

**int** year;

**int** month;

**int** day;

}Date;

Date Gcalendar;//定义一个万年历（全局变量）

**算法4.1 int main()**

作 用： 测试自己定义的函数

参 数： 无参数

返回值： 始终返回0；

计算过程：

1. 调用Get\_Date()函数输入日期。
2. 调用Operation()函数对日期进行改变操作。
3. 调用Put\_Date()函数输出改变后的日期。
4. 调用F\_week()函数判断并输出当天是星期几。

**算法3.2 void Get\_Date();void Put\_Date();**

作 用： 输入日期 输出日期

参 数： 无参数

返回值： 始终返回0；

计算过程：

1. 输入/输出万年历的日期

**算法3.3 void Operation()**

作 用： 操作日期

参 数： char ch,int n

返回值： 始终返回0；

计算过程：

1. 如果接收到的ch为‘y’，直接令万年历的year值进行自增n。
2. 如果接收到的ch为‘m’，那么调用OP\_month()函数。
3. 如果接收到的ch为‘d’，那么调用Op\_day()函数。

**算法3.4 void Op\_month()**

作 用： 对月份进行操作

参 数： int n

返回值： 始终返回0；

计算过程：

1. 定义flag\_1和flag\_2两个整数分别用于标记更改之前和更改之后月份的类型。
2. 使万年历的month值自增n，使万年历的year值自增month除以12的余数，使万年历的month值为month值除以12 的数值。
3. 使用flag\_1,flag\_2分别标记更改之前和更改之后的月份的类型。
4. 如果万年历的day值恰好为31且更改前的月份为31天月份，且更改之后月份为30天月份，令万年历的天数为30.
5. 如果更改后的月份值为2月份，且万年历的day值恰好为30或者31天（由月份决定），且万年历的年份为闰年，令万年历的day值为29，如果是平年则为28 。

**算法3.5 void Op\_day()**

作 用： 操作日期

参 数： int n

返回值： 始终返回0；

计算过程：

1. 使万年历的day值自增n
2. 判断万年历的month值的类型
3. 如果万年历的month是31天的月份，调用Op\_month()函数对万年历的day除以31的值进行操作，使万年历的day值为day值对于31的余数。
4. 如果万年历的month是30天的月份，调用Op\_month()函数对万年历的day除以30的值进行操作，使万年历的day值为day值对于30的余数。
5. 如果万年历的month是2月份并且万年历的year值是闰年，调用Op\_month()函数对万年历的day除以29的值进行操作，使万年历的day值为day值对于29的余数。
6. 如果万年历的month是2月份并且万年历的year值是平年，调用Op\_month()函数对万年历的day除以28的值进行操作，使万年历的day值为day值对于28的余数。

**算法3.6 bool If\_Leap()**

作 用： 判断是否是闰年

参 数： int year

返回值： 返回bool类型值true/false

计算过程：

1. 能够被100整除但是不能被400整除的不是闰年，返回false。
2. 不能够被4整除的不是闰年，返回false。
3. 其余的都是闰年，返回true

**算法3.7 int F\_month()**

作 用： 判断月份的类型

参 数： int n

返回值： 返回整形标记值flag。

计算过程：

1. 2月份单独标记为1 。
2. 1、3、5、7、8、10、12月份被标记为2 。
3. 其余被标记为3 。

**算法3.6 bool F\_week()**

作 用： 判断对应日期是星期几

参 数： 无参数

返回值： 返回bool型数据true/false

计算过程：

1. 定义一个time\_t类型的值timeraw。
2. 定义一个星期类型名称数组weekday。
3. 使用<ctime>文件计算与1990/1/1日的差值计算万年历当天是星期几并且输出。

**算法3.6 void F\_Monday()**

作 用： 寻找最近的一个星期一

参 数： 无参数

返回值： 始终返回0；

计算过程：

1. 当F\_week()函数返回为假时，持续使万年历的day值自加1.
2. 找到对应日期后输出日期。

# 测试用例与测试结果

## **命令行参数（选做）**

**测试用例：**

**apple banana pear**

**输出结果：**

**apple banana pear**

## **二叉树及其操作**

**测试用例：**

**5**

**pear**

**apple**

**apple**

**banana**

**banana**

**输出结果：**

**apple 2**

**banana 2**

**pear 1**

## **可变参数表**

**测试用例：**

**error("My name is zsy . \n");**

**error("%s %d %s\n","I'm",19,"year's old . ");**

**error("%c %s\n",'I',"love China ! ");**

**输出结果：**

**My name is zsy .**

**I'm 19 year's old .**

**I love China !**

## **日期计算**

**测试用例：**

**1.2016 3 30 m 1**

**2.2016 12 30 m -1**

**3.2016 12 31 m 2**

**4.2017 2 28 m 1**

**5.2016 3 31 m 1**

**6.2016 12 31 m -1**

**7.2015 12 31 m 2**

**8.2016 2 29 m 1**

**输出结果：**

**1.**

**Now date is:2016/4/30**

**That day is a Saturday**

**First Monday:**

**Now date is:2016/5/2**

**2.**

**Now date is:2016/11/30**

**That day is a Wednesday**

**First Monday:**

**Now date is:2016/12/5**

**3.**

**Now date is:2017/2/28**

**That day is a Tuesday**

**First Monday:**

**Now date is:2017/3/6**

**4.**

**Now date is:2017/3/28**

**That day is a Tuesday**

**First Monday:**

**Now date is:2017/4/3**

**5.**

**Now date is:2016/4/30**

**That day is a Saturday**

**First Monday:**

**Now date is:2016/5/2**

**6.**

**Now date is:2016/11/30**

**That day is a Wednesday**

**First Monday:**

**Now date is:2016/12/5**

**7.**

**Now date is:2016/2/29**

**That day is a Monday**

**First Monday:**

**Now date is:2016/2/29**

**8.**

**Now date is:2016/3/29**

**That day is a Tuesday**

**First Monday:**

**Now date is:2016/4/4**

# 实验总结

本次实验一共包括四个上机题，一道选做的命令行参数的题，让我对命令行的理解更加的深刻，学会了如何使用命令行给程序提供输入值，也学会了如何在win系统上使用cmd和在macOS上使用terminal运行命令行参数。第二道题是可变参数的题，学会了如何使用头文件<csdtarg>，使用宏进行获取可变参数，模拟printf()函数的功能。第三题让我又重温了二叉排序树的操作，更是使用了c++特有的new和delete进行空间的申请和释放。第四题模拟了万年历，让我明白了模块化编程的重要性，这样做大大提高了程序的可读性，使整个程序更加的简洁明了，这次上机收获颇丰。