**《（课程名称）》实验报告**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **年级、专业、班级** | | **2019级计算机科学与技术卓越02班** | | | **姓名** | **李燕琴** |
| **实验题目** | 二叉树的应用 | | | | | |
| **实验时间** | **2020/11/03** | | **实验地点** | **DS1401** | | |
| **实验成绩** |  | | **实验性质** | **□验证性 √设计性 □综合性** | | |
| 教师评价：  □算法/实验过程正确； □源程序/实验内容提交 □程序结构/实验步骤合理；  □实验结果正确； □语法、语义正确； □报告规范；  其他：  评价教师签名： | | | | | | |
| 一、实验目的  1. 学习并掌握二叉树数据结构及实现方法，掌握树的遍历方法。  2. 熟练掌握二叉检索树（BST）的基本操作及其应用。 | | | | | | |
| 二、实验项目内容   1. 利用BST实现一个城市数据库：每个数据库结点包括城市名称和以整数x与y表示的城市坐标，根据城市名称组织该BST； 2. 在该数据库上实现按城市名称进行的插入、删除和检索； 3. 打印出以指定字母打头的所有城市记录； 4. 打印出与指定点的距离在给定值之内的所有城市记录； 5. 最后提交完整的实验报告和源程序。 | | | | | | |
| 三、实验过程或算法（源程序）  1、数据结构构建：  实验涉及的数据类，可以直接按照书上的ADT编写（其中cityname设置为字符串，根据string类对“>”和“<”的操作重载，即可完成按城市名称组织BST），有继承Dictionary类的BST类，继承BinNode类的BSTNode类，Location类描述城市位置；在BST类中根据实际问题加入功能函数①printFindA（按指定字母查找相关城市），②printFindD（按指定点的指定范围查找相关城市）  2、程序主函数  输入欲写入程序的城市数目和城市信息，设定数字指令，1：新增城市数据，2：删除城市数据，3：查询城市数据，4：查询指定字母打头的所有城市，5：查询指定点(a,b)距离d的所有城市，6：列出所有城市数据，0：退出系统。对不同的指令进行相应处理即可。其中自定义功能函数：printFindA和printFindD，均采用中序方式判断处理。 | | | | | | |
| 四、实验结果及分析和（或）源程序调试过程  —————————————实验结果—————————————  请输入待测试数据库中城市的数目(N)和数据(cityname x y)：  5  beijing 1 2  tianjing 2 3  chengdu 3 1  chongqing 5 6  xian 4 5  查询指令：  1：新增城市数据  2：删除城市数据  3：查询城市数据  4：查询指定字母打头的所有城市  5：查询指定点(a,b)距离d的所有城市  6：列出所有城市数据  0：退出系统  请输入指令：1  请按格式输入"cityname x y"：  hangzhou 7 5  请输入指令：6  beijing(1,2)  chengdu(3,1)  chongqing(5,6)  hangzhou(7,5)  tianjing(2,3)  xian(4,5)  请输入指令：2  请按格式输入"cityname"：  xian  请输入指令：6  beijing(1,2)  chengdu(3,1)  chongqing(5,6)  hangzhou(7,5)  tianjing(2,3)  请输入指令：3  请按格式输入"cityname"：  hangzhou  (7,5)  请输入指令：4  请按格式输入"letter"：  b  beijing(1,2)  请输入指令：5  请按格式输入"a b r"：  3 4 2  tianjing(2,3)  请输入指令：6  beijing(1,2)  chengdu(3,1)  chongqing(5,6)  hangzhou(7,5)  tianjing(2,3)  请输入指令：0  退出系统  —————————————问题记录—————————————  ①输入问题：最开始写代码的时候，以cin的方式总是报错：  cin >> cityname >> x >> y;  中间上网查资料中了解到istringstream类可以分割空格设置输入流。  string readline;  getline(cin, readline);  istringstream readin(readline);  readin >> cityname;  readin >> x;  readin >> y;  实验结束后，整理代码又发现cin可以正常使用了，但是并没懂当时为什么cin报错且无法运行，也花费了大量的时间去寻找解决办法。  ②最开始对题意理解不透彻，在思考如何根据字符串来组织BST的问题上也花了部分精力，后来理解了string重载了“>”和“<”操作符后，上手比较容易。  ③对书上的ADT还不太熟悉，几次Debug都是因为抄写书上ADT的时候，写错字母或少括号等（后来听大佬说在学这些内容的时候就把ADT写了一遍，学到了！还是懒，练少了）。  ④如何判断找不到指定的城市数据？我在自定义功能函数中设置了变量    Flag变量，最开始没有添加static，发现会有误判的情况，加上static就表征了全局的搜索情况。  ⑥如何判断内部查找“查询无果”的情况？    即为了追求代码的完整性，想实现上述功能，查找不到相应城市输出“无满足条件的城市”    我的做法是将findhelp和find中的函数返回值类型设为E\*，判断指针是否为空即可解决问题。在实验结束后，在和大佬交流时才发现这样也可以解决了remove函数这里的“判空”问题。    （确实觉得书上的ADT也可以将返回值设为element指针，能解决很多问题）。  ⑦代码不足性：虽然整个代码功能实现尚且完整，但是之前有想到将所有输入的cityname和letter均忽略大小写，但并未在代码中实现该功能。  —————————————源程序———————————————  // #include<bits/stdc++.h>  #include<iostream>  #include<sstream>  #include<string>  #include<cmath>  using namespace std;  /\*————————————Dictionary————————————\*/  template<typename Key, typename E>  class Dictionary{  private:      void operator=(const Dictionary&){}      Dictionary(const Dictionary&){}  public:      Dictionary(){}      virtual ~Dictionary(){}      virtual void clear() = 0;      virtual void insert(const Key &k, const E &e) = 0;      virtual E remove(const Key &k) = 0;      virtual E removeAny() = 0;      virtual E\* find(const Key &k) const = 0;      virtual int size() = 0;  };  /\*————————————BinNode————————————\*/  template <typename E>  class BinNode{  public:      virtual ~BinNode(){}      virtual E &element() = 0;      virtual BinNode\* left() const = 0;      virtual BinNode\* right() const = 0;      virtual void setElement(const E &) = 0;      virtual void setLeft(BinNode \*) = 0;      virtual void setRight(BinNode \*) = 0;      virtual bool isLeaf() = 0;  };  /\*-----------BSTNode------------\*/  template<typename Key, typename E>  class BSTNode: public BinNode<E>{  private:      Key k;      E it;      BSTNode \*lc;      BSTNode \*rc;  public:      BSTNode() { lc = rc = NULL; }      BSTNode(Key K, E e, BSTNode\* l=NULL, BSTNode\* r=NULL){          k = K; it = e; lc = l; rc = r;      }      ~BSTNode(){}      Key& key() { return k; }      E& element() { return it; }      inline BSTNode\* left() const { return lc; }      inline BSTNode\* right() const { return rc; }      void setKey(const Key &K) { k = K; }      void setElement(const E &e) { it = e; }      void setLeft(BinNode<E> \*b) { lc = (BSTNode \*)b; }      void setRight(BinNode<E> \*b) { rc = (BSTNode \*)b; }      bool isLeaf() { return (lc == NULL) && (rc == NULL); }  };  /\*-----------BST------------\*/  template<typename Key, typename E>  class BST:public Dictionary<Key, E>{  private:      BSTNode<Key, E> \*root;      int nodecount;      void clearhelp(BSTNode<Key,E>\* root){          if(root == NULL) return ;          clearhelp(root->left());          clearhelp(root->right());          delete root;      }      BSTNode<Key,E>\* inserthelp(BSTNode<Key,E>\* root,const Key& k, const E& it){          if(root==NULL)              return new BSTNode<Key, E>(k, it, NULL, NULL);          if(k < root->key())              root->setLeft(inserthelp(root->left(), k, it));          else              root->setRight(inserthelp(root->right(), k, it));          return root;      }      BSTNode<Key, E> \*deletemin(BSTNode<Key, E> \*rt){          if(rt->left() == NULL)              return rt->right();          else{              rt->setLeft(deletemin(rt->left()));              return rt;          }      }      BSTNode<Key, E> \*getmin(BSTNode<Key, E> \*rt){          if(rt->left() == NULL) return rt;          else return getmin(rt->left());      }      BSTNode<Key, E> \*removehelp(BSTNode<Key, E> \*rt,const Key& k){          if(rt == NULL) return NULL;          else if(k < rt->key())              rt->setLeft(removehelp(rt->left(), k));          else if(k > rt->key())              rt->setRight(removehelp(rt->right(), k));          else{ // find the target              BSTNode<Key, E> \*temp = rt;              if(rt->left()==NULL){                  rt = rt->right();                  delete temp;              }              else if(rt->right() == NULL){                  rt = rt->left();                  delete temp;              }              else{                  BSTNode<Key, E> \*temp = getmin(rt->right());                  rt->setElement(temp->element());                  rt->setKey(temp->key());                  rt->setRight(deletemin(rt->right()));                  delete temp;              }          }          return rt;      }      // 为了方便判断数据是否存在，修改了原始数据，原来返回E，现在返回E\*      E\* findhelp(BSTNode<Key,E>\* root,const Key& k) const{          if(root == NULL) return NULL;          if(k < root->key())              return findhelp(root->left(), k);          else if(k > root->key())              return findhelp(root->right(), k);          else              return &(root->element());      }      void printhelp(BSTNode<Key,E>\* root) const{          if(root == NULL) return ;          printhelp(root->left());          cout << root->key() << root->element() <<"\n";          printhelp(root->right());      }      // 自定义函数（封装性不是很好）      int printFindAhelp(BSTNode<Key,E>\* root, char letter){          if(root==NULL) return 0;          static int flag = 0;          printFindAhelp(root->left(),letter);          if(root->key()[0] == letter){              cout << root->key() << root->element() << endl;              flag = 1;          }          printFindAhelp(root->right(),letter);          return flag;      }      int printFindDhelp(BSTNode<Key,E>\* root, int a,int b, int r){          if(root==NULL) return 0;          static int flag = 0;          printFindDhelp(root->left(),a,b,r);          int x = root->element().X();          int y = root->element().Y();          if(sqrt((x-a)\*(x-a)+(y-b)\*(y-b)) < r){              cout << root->key() << root->element() << endl;              flag = 1;          }          printFindDhelp(root->right(),a,b,r);          return flag;      }  public:      BST() { root = NULL; nodecount = 0; }      ~BST() { clearhelp(root); }      void clear() { clearhelp(root); root=NULL; nodecount=0;}      void insert(const Key &k, const E &e) {          root = inserthelp(root, k, e);          nodecount++;      }      E remove(const Key& k){          E\* temp = findhelp(root, k);          if(temp != NULL){              root = removehelp(root, k);              nodecount--;          }          return \*temp;      }      E removeAny(){          if(root!=NULL){              E temp = root->element();              root = removehelp(root, root->key());              nodecount--;              return temp;          }          else return NULL;      }      E\* find(const Key &k) const { return findhelp(root, k); }      int size() { return nodecount; }      void print() const{          if(root==NULL) cout<<"The BST is empty.\n";          // else printhelp(root,0);          else printhelp(root);      }      void printFindA(char letter){          if(printFindAhelp(root,letter)==0) {              cout << "无满足条件的城市" << endl;          }      }      void printFindD(int a,int b, int r){          if(printFindDhelp(root, a, b, r)==0) {              cout << "无满足条件的城市" << endl;          }      }  };  /\*------------Location------------\*/  class Location{      friend ostream& operator<<(ostream &out, Location A){          out << "(" << A.X() << "," << A.Y() << ")" ;          return out;      }      friend istream& operator>>(istream &input, Location A){          int x, y;          input >> x >> y;          A.setX(x);          A.setY(y);          return input;      }  private:      int x;      int y;  public:      Location(int a=-1, int b=-1) { x = a; y = b;}      ~Location(){}      int X() { return x; }      int Y() { return y; }      void setX(int a) { x = a; }      void setY(int b) { y = b; }      bool operator==(Location &A){          return (x == A.X() && y == A.Y());      }      bool operator!=(Location &A){          return (x != A.X() || y != A.Y());      }      // void operator=(Location &A){      //     x = A.X(); y=A.Y();      // }  };  int main(){      BST<string, Location> CityData;      int N, x, y;      string cityname;      cout << "请输入待测试数据库中城市的数目：" ;      scanf("%d\n", &N);      cout << "请按格式输入城市信息\"cityname x y\"：" << endl;      for (int i = 0; i < N; i++){          cin >> cityname >> x >> y;          Location \*cityXY = new Location(x,y);          CityData.insert(cityname, \*cityXY);          cityname = "";          delete cityXY;          cityXY = NULL;      }      cout << "查询指令：\n"           << "1：新增城市数据\n"           << "2：删除城市数据\n"           << "3：查询城市数据\n"           << "4：查询指定字母打头的所有城市\n"           << "5：查询指定点(a,b)距离d的所有城市\n"           << "6：列出所有城市数据\n"           << "0：退出系统\n";      // getchar();      while(true){          int cmd;          cout << "请输入指令：";          scanf("%d", &cmd);          if(cmd == 1){              getchar();              cout << "请按格式输入\"cityname x y\"：" << endl;              cin >> cityname >> x >> y;              Location \*cityXY = new Location(x,y);              CityData.insert(cityname, \*cityXY);              cityname = "";              delete cityXY;              cityXY = NULL;          }          else if(cmd == 2 ){              getchar();              cout << "请按格式输入\"cityname\"：" << endl;              cin >> cityname;              CityData.remove(cityname);              cityname = "";          }          else if(cmd == 3){              getchar();              cout << "请按格式输入\"cityname\"：" << endl;              cin >> cityname;              Location \*temp = CityData.find(cityname);              if(temp==NULL){                  cout << "无满足条件的城市" << endl;              }              else cout << \*temp << endl;              cityname = "";          }          else if(cmd == 4){              char letter;              getchar();              cout << "请按格式输入\"letter\"：" << endl;              cin >> letter;              CityData.printFindA(letter);          }          else if(cmd == 5){              int a, b, r;              cout << "请按格式输入\"a b r\"：" << endl;              cin >> a >> b >> r;              CityData.printFindD(a, b, r);          }          else if(cmd == 6){              CityData.print();          }          else if(cmd == 0){              cout << "退出系统" << endl;              break;          }          else {              cout << "输入有误，请重新输入" << endl;          }      }      return 0;  } | | | | | | |

备注：

1、教师在布置需撰写实验报告的实验前，应先将报告书上的“实验题目”、“实验性质”、“实验目的”、“实验项目内容”等项目填写完成，然后再下发给学生。

2、教师在布置需撰写报告的实验项目时，应告知学生提交实验报告的最后期限。

3、学生应按照要求正确地撰写实验报告：

* 1. 在实验报告上正确地填写“实验时间”、“实验地点”等栏目。
  2. 将实验所涉及的源程序文件内容（实验操作步骤或者算法）填写在“实验过程或算法（源程序）”栏目中。
  3. 将实验所涉及源程序调试过程（输入数据和输出结果）或者实验的分析内容填写在“实验结果及分析和（或）源程序调试过程”栏目中。
  4. 在实验报告页脚的“报告创建时间：”处插入完成实验报告时的日期和时间。
  5. 学生将每个实验完成后，按实验要求的文件名通过网络提交（上载）到指定的服务器所规定的共享文件夹中。每个实验一个电子文档，如果实验中有多个电子文档（如源程序或图形等），则用WinRAR压缩成一个压缩包文档提交，压缩包文件名同实验报告文件名（见下条）。
  6. 提交的实验报告电子文档命名为：“年级（两位数字不要“级”字）专业（缩写：计算机科学与技术专业（计科）、网络工程专业（网络）、信息安全专业（信息）、物联网工程（物联网））班级（两位数字）学号（八位数字）姓名实验序号（一位数字）．doc。如学号为20115676、年级为2011级、专业为“计算机科学与技术”专业、班级为“02班”、姓名为“王宇”的学生，完成的第一次实验命名为： 11计科02班20115676王宇1．Doc，以后几次实验的报告名称以此类推。

4、教师（或助教）在评价学生实验时，应根据其提交的其他实验相关资料（例如源程序文件等）对实验报告进行仔细评价。评价后应完成的项目有:

1. 在“成绩”栏中填写实验成绩。每个项目的实验成绩按照总分10评分。
2. 在“教师评价”栏中用符号标注评价项目结果（用√表示正确，用×表示错误，用≈表示 半对半错）。
3. 在“教师评价”栏中“评价教师签名”填写评价教师（或助教）姓名。将评价后的实验报告转换为PDF格式文件归档。
4. 课程实验环节结束后，任课教师将自己教学班的实验报告文件夹进行清理。在提交文件夹中，文件总数为实验次数×教学班学生人数（如，教学班人数为90人，实验项目为5，其文件数为：90×5=450）。任课教师一定要认真清理，总数相符，否则学生该实验项目不能得分。最后将学生提交的实验报告刻光盘连同实验成绩一起放入试卷袋存档。