四 川 大 学 计 算 机 学 院、软 件 学 院

实 验 报 告

学号：2017141461297 姓名：郭安洁 专业：计算机科学 班级：＿＿＿＿ 第 6 周

|  |  |  |  |
| --- | --- | --- | --- |
| 课程名称 | 数据结构与算法分析课程设计 | 实验课时 | 4 |
| 实验项目 | 实验一:带括号的算术表达式求值 | 实验时间 | 2018.10.10~2018.10.17 |
| 实验目的 | 1.采用算符优先数算法,能正确求值表达式;  2.熟练掌握栈的应用;  3.熟练掌握计算机系统的基本操作方法,了解如何编辑、编译、链接和运行一个C++程序;  4.上机调试程序,掌握查错、排错使程序能正确运行。 | | |
| 实验环境 | 软件环境：操作系统：Windows10； | | |
| 实验内容（算法、程序、步骤和方法） | **算法描述**：  在solve函数中处理字符串表达式；  每个运算符根据其优先顺序有一个级别，越优先者级别越高；  运算符用结构体Op表示，用于保存该运算符内容及级别；  用字符串保存输入的算术表达式，依次读取每个字符，判断：若该字符为数字，则计算出该数字段的数值（包括小数），压入数据栈num；  若该字符为运算符（记为x），则将该运算符级别与上一个运算符（记为y）级别比较，若该字符级别高，则将其压入运算符栈op，反之，则先计算上一个运算符（y）所在的表达式，从num中弹出两个数据，先后分别记为b和a，计算ayb表达式并将结果压入num，从op弹出y，压入x；  若该字符为(，则所有运算符级别都增加5（至少高于增加前的最高级别）；若该字符为)，所有运算符级别都减少5；  直到读入完成并且op中所有元素都弹出（处理完成），将会一直循环；  运算结果为num的栈顶元素。  **注**：该程序可处理带小数、+、-、\*、/、^、%、括号的算术表达式，且一个式子中没有空格；  可在处理字符串时判断是否有错误输入（表达式中有字母、除法和模运算中被除数为零、数据有前导小数点、数值在小数点前有多个前导零），若有，则solve函数返回false。 | | |
| （接上）  实验内容（算法、程序、步骤和方法） | 程序清单：   1. #include <iostream> 2. #include <vector> 3. #include <stack> 4. #include <cstring> 5. #include <string> 6. #include <cctype> 7. #include <cmath> 8. #define MIN 5 9. #define LOCAL 10. **using** **namespace** std; 12. **struct** Op{ 13. **char** name; 14. **int** level; 15. Op(**char** n):name(n){ 16. **if**(n=='+'){ 17. level=1; 18. } 19. **else** **if**(n=='-'){ 20. level=2; 21. } 22. **else** **if**(n=='\*'||n=='/'||n=='%'){ 23. level=3; 24. } 25. **else** **if**(n=='^'){ 26. level=4; 27. } 28. } 29. **char** getOp(){ 30. **return** name; 31. } 32. **void** opAddLevel(**int** addLevel){ 33. **this**->level+=addLevel; 34. } 35. }; 37. **bool** toDouble(**char** s[],**int**& i,**double**& ans){ 38. **double** temp=1; 39. **int** isIntPart=1,startWithZero=0; 40. **while**(isdigit(s[i])||s[i]=='.'){ 41. **if**(isdigit(s[i])){ 42. **if**(s[i]=='0'){ 43. **if**(startWithZero==0){ 44. startWithZero=1; 45. } 46. **else** **if**(startWithZero&&isIntPart){ 47. **return** **false**; 48. } 49. } 50. **if**(isIntPart==1){ 51. ans\*=10; 52. ans+=s[i]-'0'; 53. } 54. **else**{ 55. temp/=10; 56. ans+=temp\*(s[i]-'0'); 57. } 58. } 59. **else** **if**(s[i]=='.'){ 60. isIntPart=0; 61. } 62. i++; 63. } 65. **return** **true**; 66. }  69. **bool** opTwoNums(stack<**double**>& num,**char** op,**double**& temp\_ans){ 70. **double** b=num.top();num.pop(); 71. **double** a=num.top();num.pop(); 72. **if**(op=='+'){ 73. temp\_ans=a+b; 74. } 75. **else** **if**(op=='-'){ 76. temp\_ans=a-b; 77. } 78. **else** **if**(op=='\*'){ 79. temp\_ans=a\*b; 80. } 81. **else** **if**(op=='/'){ 82. **if**(b==0){ 83. **return** **false**; 84. } 85. temp\_ans=a/b; 86. } 87. **else** **if**(op=='%'){ 88. **if**(b==0){ 89. **return** **false**; 90. } 91. temp\_ans=(**int**)a%(**int**)b; 92. } 93. **else** **if**(op=='^'){ 94. temp\_ans=pow((**int**)a,(**int**)b); 95. } 96. **return** **true**; 97. } 99. **bool** solve(**char** s[],**double**& ans){ 100. **int** len=strlen(s); 101. stack<**double**> num; 102. stack<Op> op; 103. **int** addLevel=0,i=0; 105. **while**(i<len||op.size()>0){//还在读入 或 已经读入完成并还在处理 106. **if**(i<len){ 107. **if**(isdigit(s[i])){ 108. **double** temp\_int=0; 109. **if**(!toDouble(s,i,temp\_int)){ 110. **return** **false**; 111. } 112. num.push(temp\_int); 113. } 114. **else** **if**(s[i]=='('){ 115. addLevel+=MIN; 116. i++; 117. } 118. **else** **if**(s[i]==')'){ 119. addLevel-=MIN; 120. i++; 121. } 122. **else** **if**(isalpha(s[i])||s[i]=='.'){ 123. **return** **false**; 124. } 126. **else**{  //if s[i] is an operator 127. **if**(s[i]=='-'){ 128. **if**(i==0||(i>0&&!isdigit(s[i-1]))){ 129. num.push(0); 130. } 131. } 132. Op s\_i(s[i]); 133. s\_i.opAddLevel(addLevel); 134. **if**(!op.empty()){ 135. Op op\_top=op.top(); 136. **if**(op\_top.level>=s\_i.level){ 137. op.pop(); 138. **double** temp\_ans; 139. **if**(!opTwoNums(num,op\_top.getOp(),temp\_ans)){ 141. **return** **false**; 142. } 143. num.push(temp\_ans); 144. } 145. **else**{ 146. op.push(s\_i);i++; 147. } 148. } 149. **else**{ 150. op.push(s\_i);i++; 151. } 152. } 153. } 155. **else**{ 156. **double** temp\_ans; 157. **if**(!opTwoNums(num,op.top().getOp(),temp\_ans)){ 158. **return** **false**; 159. } 160. op.pop(); 161. num.push(temp\_ans); 162. } 164. **if**(!num.empty()){ 165. ans=num.top(); 166. } 167. } 169. **return** **true**; 170. }  173. **int** main(){ 174. #ifdef LOCAL 175. freopen("input.txt","r",stdin); 176. #endif 178. **char** s[100]; 179. **while**(cin>>s){ 180. **double** ans=0; 181. **if**(solve(s,ans)){ 182. cout<<s<<"="<<ans<<endl; 183. } 184. **else**{ 185. cout<<s<<" is an invalid input"<<endl; 186. } 187. } 189. **return** 0; 190. } | | |
| 数据记录  和计算 |  | | |
| 结 论  （结 果） |  | | |
| 小 结 |  | | |
| 指导老师评 议 | 成绩评定： 指导教师签名： | | |

实验报告说明

专业实验中心

**实验名称** 要用最简练的语言反映实验的内容。如验证某程序、定律、算法，可写成“验证×××”；分析×××。

**实验目的** 目的要明确，要抓住重点，可以从理论和实践两个方面考虑。在理论上，验证定理、公式、算法，并使实验者获得深刻和系统的理解，在实践上，掌握使用实验设备的技能技巧和程序的调试方法。一般需说明是验证型实验还是设计型实验，是创新型实验还是综合型实验。

**实验环境** 实验用的软硬件环境（配置）。

**实验内容（算法、程序、步骤和方法）** 这是实验报告极其重要的内容。这部分要写明依据何种原理、定律算法、或操作方法进行实验，要写明经过哪几个步骤。还应该画出流程图（实验装置的结构示意图），再配以相应的文字说明，这样既可以节省许多文字说明，又能使实验报告简明扼要，清楚明白。

**数据记录和计算** 指从实验中测出的数据以及计算结果。

**结论（结果）** 即根据实验过程中所见到的现象和测得的数据，作出结论。

**小结** 对本次实验的体会、思考和建议。

**备注或说明** 可写上实验成功或失败的原因，实验后的心得体会、建议等。

**注意：**

* 实验报告将记入实验成绩；
* 每次实验开始时，交上一次的实验报告，否则将扣除此次实验成绩。