

GF3002

脚本语言

刘志伟

刘志伟

C与LUA交互之数组操作与字符串处理

实验1

余庆祥

2220631136

BIH

2022/11/27

|  |  |
| --- | --- |
| **1、实验名称** | **实验4 C与LUA交互之数组操作与字符串处理** |
| **2、实验日期** | **2022/11/27** |
| **3、实验目的** | |
| 1. 理解迭代器的概念。 2. 掌握迭代器的编写。 3. 掌握泛型for的原理及应用。 | |
| **4、实验环境** | |
| win7以上兼容系统，lua解释器，VS2019 | |
| **5、实验的步骤和方法** | |
| **(将你设计程序的步骤详细描述，例如关键代码编写顺序说明，设计出来的类图，相关类成员的说明等等)**   1. 打开vs2019，创建控制台项目。创建test.lua文件，并在其中定义一个数组array = {"Hello", 1, "World", 23.2}。编写C++程序，使用lua\_rawgeti函数实现对数组array的调用。 2. 在lua文件中定义一个空数组array1 = { }，在C++中使用lua\_rawseti函数实现向数组array1中写入整数数据。 3. 在test.lua中添加如下语句。 4. 补充下面空缺的代码，并添加到C++程序中。 5. 在项目中用C++语言为lua脚本编写一个函数实现字符串大小写切换的功能。若lua中此功能的函数取名为 toggle ，请实现此函数并进行验证。 6. 编写一个简单的统计库(Statist).实现对一个数字型数组进行以下统计功能。 | |
| **6、数据记录和计算** | |
| **（代码编写思路及关键代码）**   1. 第一题关键代码   void readLuaArray(lua\_State\* L, const char\* arrayName)  {  lua\_getglobal(L, arrayName); // 指定的数组入栈  luaL\_checktype(L, -1, LUA\_TTABLE); // 检查是否是table  int n = (int)luaL\_len(L, -1); // 获得该table的元素个数  for (int i = 1; i <= n; i++) // table 下标从1开始  {  lua\_rawgeti(L, -1, i); // 取i下标的元素入栈  cout << arrayName << '[' << i << "]: " << lua\_tostring(L, -1) << endl;  lua\_pop(L, 1);  }  }   1. 第二题关键代码：   void writeLuaArray(lua\_State\* L, const char\* arrayName)  {  lua\_getglobal(L, arrayName);  luaL\_checktype(L, -1, LUA\_TTABLE);  int n = (int)luaL\_len(L, -1);  for (int i = 1; i <= 3; i++)  {  lua\_pushnumber(L, 10 \* i); // value 入栈  lua\_rawseti(L, -2, i + n);  }  }   1. 第三四题代码：   // lua数组变换  int CforLUA\_arrayTransformation(lua\_State\* L)  {  // 第一个参数必须是table(t)  luaL\_checktype(L, 1, LUA\_TTABLE);  // 第二个参数必须是函数(f)  luaL\_checktype(L, 2, LUA\_TFUNCTION);  // 获取table的大小  int n = (int)luaL\_len(L, 1);  for (int i = 1; i <= n; i++)  {  // 压入f => 将要调用的函数入栈 // f -> f(t[i]) -> return f(t[i]) -> new t[i]  lua\_pushvalue(L, 2);  // 压入t[i] => 将要使用的实参入栈  lua\_rawgeti(L, 1, i);  // 调用f(t[i]) => 函数调用  lua\_call(L, 1, 1);  // t[i] = 结果 => 修改数组元素的值  lua\_rawseti(L, 1, i);  }  return 0; // 无返回值  }  // 求lua数组平均值  static int CforLUA\_AVG(lua\_State\* L)  {  luaL\_checktype(L, 1, LUA\_TTABLE); // 检查栈底是否是table类型  int n = (int)luaL\_len(L, 1); // 获取table的长度  double sum = 0;  for (int i = 1; i <= n; i++)  {  lua\_pushinteger(L, i); // 将table的key值入栈  lua\_gettable(L, -2); // 将table的key值出栈，到-2的table寻找value，再将value入栈  sum += lua\_tonumber(L, -1); // 读-1栈，即上一步执行后的value  lua\_pop(L, 1); // 删除上一步指向的value  }  lua\_pushnumber(L, sum / n);  return 1;  }  // 求lua数组平均值，使用lua\_rawgeti(L, t, key)  static int CforLUA\_AVGRAW(lua\_State\* L)  {  luaL\_checktype(L, 1, LUA\_TTABLE);  int n = (int)luaL\_len(L, 1);  double sum = 0;  for (int i = 1; i <= n; i++)  {  lua\_rawgeti(L, 1, i);  sum += lua\_tonumber(L, -1);  lua\_pop(L, 1);  }  lua\_pushnumber(L, sum / n);  return 1;  }  // 字符串分割  int CforLUA\_split(lua\_State\* L)  {  const char\* pSrc = luaL\_checkstring(L, 1);  const char\* pSep = luaL\_checkstring(L, 2);  const char\* pSepLocation = NULL;  lua\_newtable(L); // 在虚拟栈创建空表，用于存放结果  int i = 1;  while ((pSepLocation = strchr(pSrc, \*pSep)) != NULL)  {  lua\_pushlstring(L, pSrc, pSepLocation - pSrc);  lua\_rawseti(L, -2, i++);  pSrc = pSepLocation + 1;  }  lua\_pushstring(L, pSrc);  lua\_rawseti(L, -2, i);  return 1; // 1个返回值，即整个数组  }  // 转换为大写字母  int CforLUA\_upper(lua\_State\* L)  {  size\_t len;  luaL\_Buffer buf;  const char\* s = luaL\_checklstring(L, 1, &len);  luaL\_buffinit(L, &buf);  for (size\_t i = 0; i < len; i++)  {  luaL\_addchar(&buf, toupper((unsigned char)(s[i])));  }  luaL\_pushresult(&buf);  return 1;  }  int CforLUA\_lower(lua\_State\* L)  {  size\_t len;  luaL\_Buffer buf;  const char\* s = luaL\_checklstring(L, 1, &len);  luaL\_buffinit(L, &buf);  for (size\_t i = 0; i < len; i++)  {  luaL\_addchar(&buf, tolower((unsigned char)(s[i])));  }  luaL\_pushresult(&buf);  return 1;  }  int CforLUA\_change(lua\_State\* L)  {  size\_t len;  luaL\_Buffer buf;  const char\* s = luaL\_checklstring(L, 1, &len);  luaL\_buffinit(L, &buf);  for (size\_t i = 0; i < len; i++)  {  if(s[i] >= 65 && s[i] < 90)  luaL\_addchar(&buf, tolower((unsigned char)(s[i])));  else if(s[i] >= 97 && s[i] < 122)  luaL\_addchar(&buf, toupper((unsigned char)(s[i])));  else  luaL\_addchar(&buf, (unsigned char)(s[i]));  }  luaL\_pushresult(&buf);  return 1;  }   1. 第五题关键代码：   int CforLUA\_change(lua\_State\* L)  {  size\_t len;  luaL\_Buffer buf;  const char\* s = luaL\_checklstring(L, 1, &len);  luaL\_buffinit(L, &buf);  for (size\_t i = 0; i < len; i++)  {  if (s[i] >= 65 && s[i] < 90)  luaL\_addchar(&buf, tolower((unsigned char)(s[i])));  else if (s[i] >= 97 && s[i] < 122)  luaL\_addchar(&buf, toupper((unsigned char)(s[i])));  else  luaL\_addchar(&buf, (unsigned char)(s[i]));  }  luaL\_pushresult(&buf);  return 1;  }  lua\_register(pluaState, "toggle", CforLUA\_change); | |
| **7、实验结果或结论（**实验结果怎么样？你从这个实验你学会了什么？得出了什么结论？） | |
| **（实验结果截图+文字说明）**   1. 第一题效果图      1. 第二题效果图      1. 第三四题效果图      1. 第五题效果图 | |
| **8、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。）**  **学会了lua与C++之间的函数通信，更灵活的使用C++和lua** | |