

《数据结构课程设计》

**实验报告**

**实验名称：顺序表**

**姓 名：唐子辉**

**学 号：19081630**

**专 业：通信工程**

**实验时间：2021年4月2日**

**杭州电子科技大学**

**通信工程学院**

**（实验报告要求：填写三四五一共三部分内容，实验报告文件命名方式：*数据结构实验报告1-姓名-学号*，完成后上传电子版到校园网的网络教学平台，无需打印。上传截止日期：4月1日24:00）**

1. **实验目的**

掌握顺序表的基本算法设计。

1. **实验内容**
2. 熟悉C语言编译环境（推荐CODEBLOCKS）；
3. 完成《数据结构教程第5版》（或《上机指导》）上机实验题2.1

\*要求：对照代码后的注释，理解程序；

1. 完成试验任务书中的指定附加题
2. **算法设计**

书上函数设计：

*void* createList(SqList \*&L,ElemType a[],*int* n){  
 L=(SqList \* )malloc(*sizeof* (SqList));  
 *for* (*int* i=0;i<n;i++){  
 L->data[i]=a[i];  
 }  
 L->length=n;  
}  
  
*void* initList(SqList \*&L){  
 L=(SqList\*)malloc(*sizeof*(SqList));  
 L->length=0;  
}  
  
*void* destroyList(SqList \*&L){  
 free(L);  
}  
  
*bool* isEmpty(SqList \*L){  
 *return* (L->length==0);  
}  
  
*int* getListLength(SqList \*L){  
 *return* (L->length);  
}  
  
*void* dispList(SqList \*L){  
 *for* (*int* i=0;i<L->length;i++){  
 printf("%c",L->data[i]);  
 }  
 printf("\n");  
}  
  
*bool* getElem(SqList \*L,*int* i,ElemType &e){  
 *if* (i<1 || i>L->length){  
 *return false*;  
 }  
 e=L->data[i-1];  
 *return true*;  
}  
  
*int* getLocateElem(SqList \*L,ElemType e){  
 *int* i=0;  
 *while* (i<L->length&&L->data[i]!=e){  
 i++;  
 }  
 *if* (i>=L->length){  
 *return* 0;  
 } *else*{  
 *return* i+1;  
 }  
}  
  
*bool* insertElem(SqList \*L,*int* i,ElemType e){  
 *int* j;  
 *if*(i<1 || i>L->length+1){  
 *return false*;  
 }  
 i--;  
 *for* (j=L->length;j>i;j--){  
 L->data[j]=L->data[j-1];  
 }  
 L->data[i]=e;  
 L->length++;  
 *return true*;  
}  
  
*bool* deleteElem(SqList \*&L,*int* i,ElemType &e){  
 *int* j;  
 *if* (i<1 || i>L->length){  
 *return false*;  
 }  
 i--;  
 e=L->data[i];  
 *for* (j=i; j < L->length-1; j++) {  
 L->data[j]=L->data[j+1];  
 }  
 L->length--;  
 *return true*;  
}

1. 实验1中，在main()函数中加入代码，调用基本运算CreateList()创建顺序表；

代码：

SqList \*N;  
*char* a[5]={'q','w','e','r','d'};  
createList(N, *reinterpret\_cast*<ElemType \*>(a), 5);  
dispList(N);

结果：

qwerd

1. 给出附加题的算法代码。
2. 附加题1

代码：

定义函数deleteRepeatedElem

*void* deleteRepeatedElem(SqList \*&L){  
 *int* slow=0;  
 *int* fast=0;  
 *int* count=0;  
 *for* (;fast<L->length;fast++) {  
 *if* (L->data[fast + 1] > L->data[fast]) {  
 L->data[slow]=L->data[fast];  
 slow++;  
 } *else*{  
 count++;  
 }  
 }  
 L->length-=count;  
}

在main函数中调用测试

SqList \*M;  
initList(M);  
insertElem(M,1,'0');  
insertElem(M,2,'0');  
insertElem(M,3,'2');  
insertElem(M,4,'4');  
insertElem(M,5,'4');  
insertElem(M,6,'4');  
insertElem(M,7,'7');  
insertElem(M,8,'7');  
dispList(M);  
deleteRepeatedElem(M);  
dispList(M);  
printf("%d\n",getListLength(M));

结果：

00244477  
0247  
4

1. 附加题2

(1&2)代码：

*int* var1;  
printf("var1 : %p\n", &var1 );  
*char* var2[5];  
*for*(*int* i=0; i<2; i++) {  
 printf("char\_var2[%d] : %p\n", i, &var2[i]);  
}  
*int* var3[5];  
*for*(*int* i=0; i<2; i++) {  
 printf("int\_var3[%d] : %p\n", i, &var3[i]);  
}  
*float* var4[5];  
*for*(*int* i=0; i<2; i++) {  
 printf("float\_var4[%d] : %p\n", i, &var4[i]);  
}  
*double* var5[5];  
*for*(*int* i=0; i<2; i++) {  
 printf("double\_var5[%d] : %p\n", i, &var5[i]);  
}

(1&2)结果：

var1 : 000000000061fddc  
char\_var2[0] : 000000000061fdd7  
char\_var2[1] : 000000000061fdd8  
int\_var3[0] : 000000000061fdc0  
int\_var3[1] : 000000000061fdc4  
float\_var4[0] : 000000000061fda0  
float\_var4[1] : 000000000061fda4  
double\_var5[0] : 000000000061fd70  
double\_var5[1] : 000000000061fd78

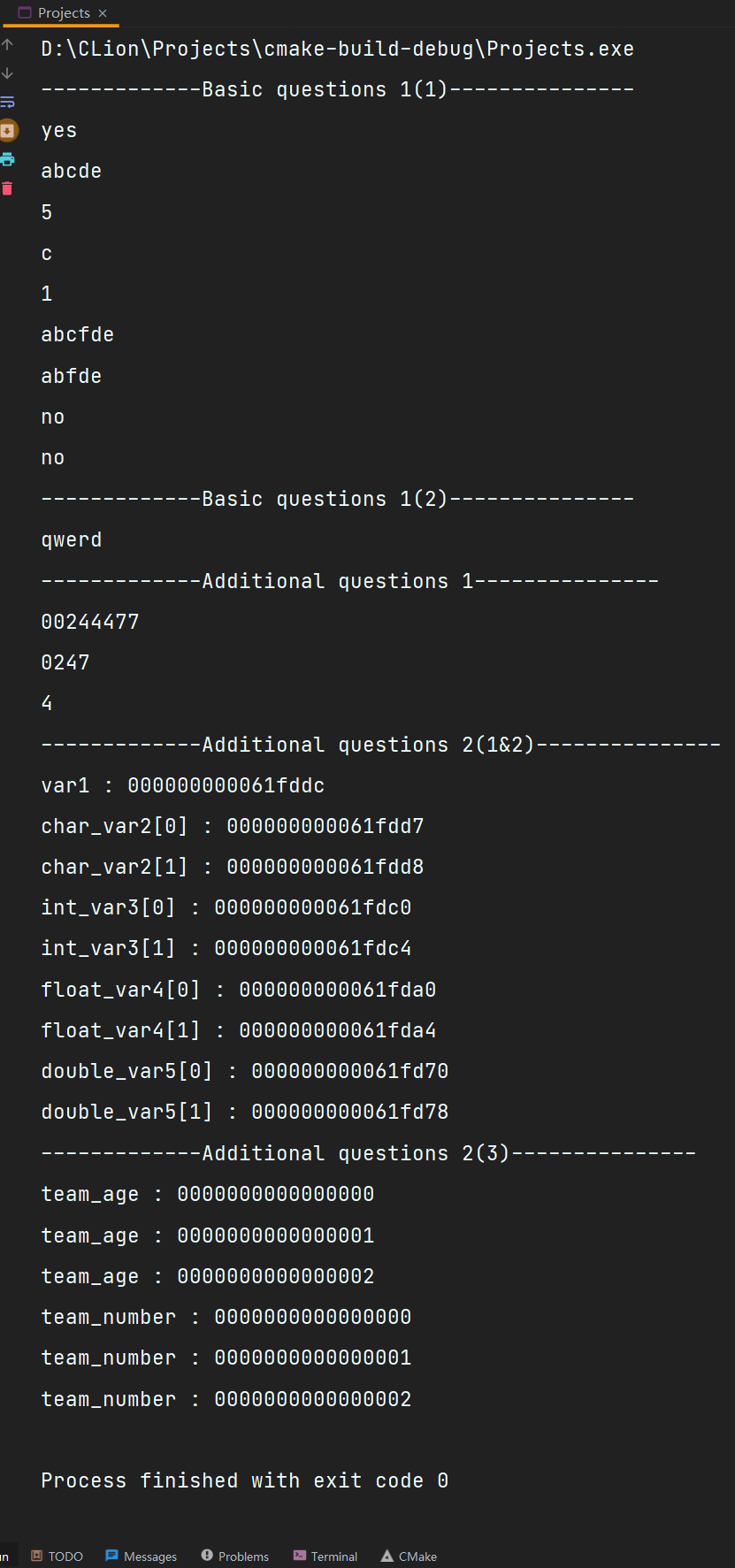
(3)代码：

*struct* player{  
 *int* number;  
 *int* age;  
}team[3];  
*for* (*int* i=0;i<3;i++){  
 printf("team\_age : %p\n", i, &team[i].age);  
}  
*for* (*int* i=0;i<3;i++){  
 printf("team\_number : %p\n", i, &team[i].number);  
}

(3)结果：

team\_age : 0000000000000000  
team\_age : 0000000000000001  
team\_age : 0000000000000002  
team\_number : 0000000000000000  
team\_number : 0000000000000001  
team\_number : 0000000000000002

1. **运行结果与分析（给出各实验题目的运行结果截图，如有必要，请加以简要说明）**



1. **思考题**：对照实验指导书，各个基本运算的函数申明中的引用运算符&的作用是什么？是否都是必要的？
2. **实验小结（记录实验过程中遇到的主要问题和心得）**

第一次写代码的时候，写在.c文件中，发现取地址符&都不能用，然后就很郁闷以为自己配置错了。后来才知道要写在.cpp文件中，然后一切问题都解决了。此外，在我调用destroyList(L)释放线性表后，再次对它进行判空操作，发现竟然不为空。然后我就测试它的长度，发现仍为5，这里感觉有点奇怪。最后就是结构体数组的内存分配，发现一个结构体中的元素用的是同一块内存，递增之后内存只加一个字节。