# 第十章 构造数据类型及其应用

本章学习目标：

* 掌握枚举类型、结构体类型、联合体类型的定义
* 了解枚举类型、结构体类型、联合体类型及其变量的区别
* 了解枚举类型、结构体类型、联合体类型的使用背景
* 掌握枚举成员的编号规则
* 掌握结构体、联合体在内存中的存储规则
* 掌握结构体数组的使用。
* 掌握链表的操作细节。

## 11.1 实践题

**一、链表的构造及遍历**

**实验目的**

1. 理解链表的概念。
2. 掌握链表的创建。
3. 掌握链表的遍历及其元素访问方法。

**实验步骤**

步骤1：略

步骤2：略

**实验结果/结论**

1. **实验结果**

* 略

2. **实验结论**

* 略

**二、 结构体数组**

**实验目的**

1. 掌握结构体的概念
2. 掌握结构体变量的使用
3. 掌握结构体指针的使用
4. 掌握结构体数组的使用
5. 掌握枚举类型的使用

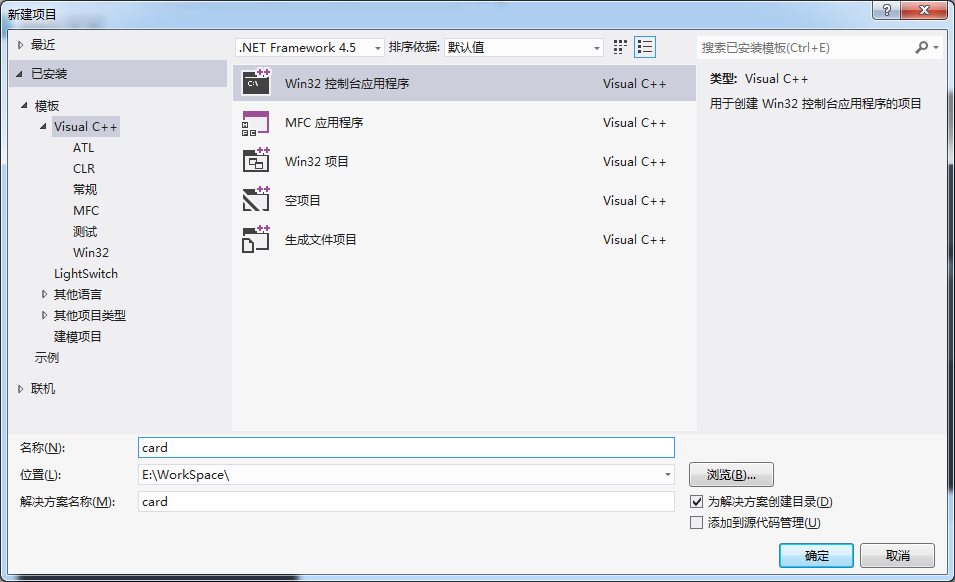
**实验步骤**

**问题描述：**

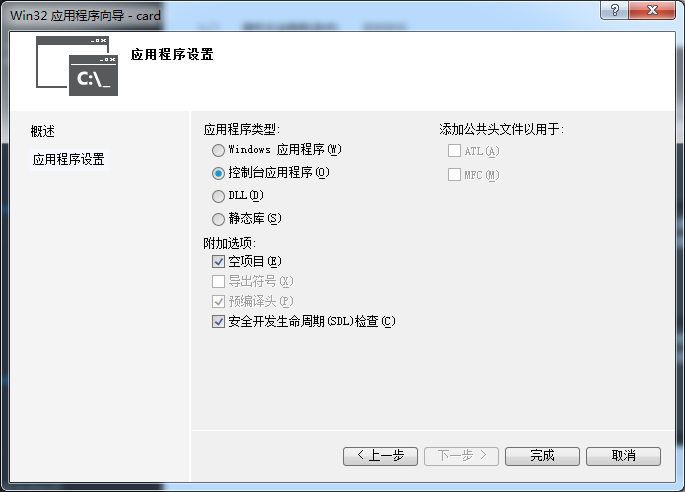
设计一个程序，模拟扑克牌游戏斗地主中在每局游戏开始时对玩家进行发牌，对本程序有如下要求及说明：

一副扑克牌总共54张牌，每名玩家17张牌，留3张作为底牌，每张扑克牌大小顺序为3、4、5、6、7、8、9、10、J、Q、K 、A、2、小王、大王；除小王、大王之外的牌都分有四种花色，花色顺序为红桃、黑桃、方块、梅花；每张牌都是唯一存在的，发牌时不能出现重复发牌，给玩家发牌结束后需要对玩家手头的牌按大小及花色进行排序。

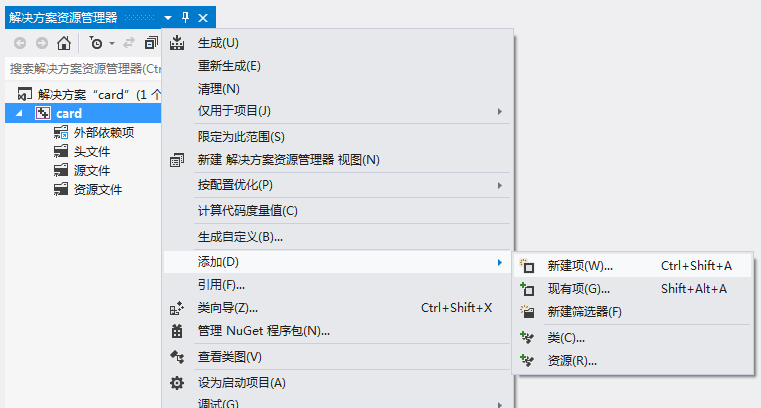
**步骤1**：打开VS2012，依次点击“文件”->“新建”->“项目”，在弹出的“新建项目”对话框中选择“Win32控制台应用程序”。



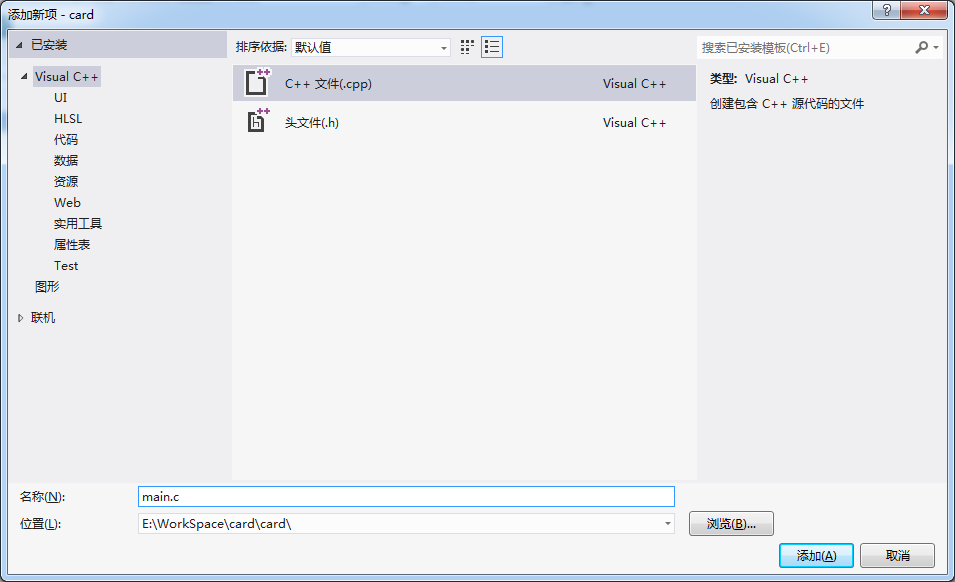
**步骤2**：在“新建项目”对话框中点击“确定”按钮进入“Win32应用程序向导”界面，点击“下一步”按钮，在“附加选项”中选择“空项目”，点击“完成”按钮，项目创建完成。



**步骤3**：在“解决方案资源管理器”中选中刚才创建的解决方案点击鼠标右键，在弹出的右键菜单中依次选择“添加”->“新建项”，弹出“添加新项”窗口。



**步骤4**：在“添加新项”窗口选择“Visual C++”->“C++文件(c.pp)”，在名称栏中输入文件名为“main.c”，点击“添加”按钮完成添加。



**步骤5**：在新创建的“main.c”文件中写入下列代码。

#include <stdio.h>

#include <stdlib.h>

#include <time.h>

// 牌面值枚举类型（3、4、5、6、7、8、9、10、J、Q、K、A、2、小王、大王）

typedef enum cardNumber

{

CardNum3 = 0, // 3

CardNum4, // 4

CardNum5, // 5

CardNum6, // 6

CardNum7, // 7

CardNum8, // 8

CardNum9, // 9

CardNum10, // 10

CardNumJ, // J

CardNumQ, // Q

CardNumK, // K

CardNumA, // A

CardNum2, // 2

CardNumKA, // 小王

CardNumKB // 大王

} CardNnumber;

// 牌面花色枚举类型（红桃、黑桃、方块、梅花、无）

typedef enum cardColor

{

Heart = 0, // 红桃

Spade, // 黑桃

Diamond, // 方块

Club, // 梅花

None // 无花色（大王、小王）

} CardColor;

// 扑克牌枚举类型

typedef struct card

{

CardColor color; // 牌面花色

CardNnumber number; // 牌面值

} Card;

void initAllCard(Card \* pCard);

void touchCard(Card \* pPlayer, int nCount, Card \* pAllCard, int \* pIsUsed);

int maxCard(Card \* pCard1, Card \* pCard2);

void sortCard(Card \* pCard, int nCount);

void printCard(Card \* pCard, int nCount);

char \* arrStrColor[] = {"红桃", "黑桃", "方块", "梅花"};

char \* arrStrNumber[] = {"3", "4", "5", "6", "7", "8", "9", "10", "J", "Q", "K", "A", "2", "小王", "大王" };

int main(int argc, char \*\* argv)

{

Card arrCardAll[54]; // 全部扑克牌

Card arrCardPlayer1[17]; // 玩家一分到的扑克牌

Card arrCardPlayer2[17]; // 玩家二分到的扑克牌

Card arrCardPlayer3[17]; // 玩家三分到的扑克牌

Card arrCardHand[3]; // 底牌

int arrIsUsed[54] = {0}; // 记录扑克牌是否已经分出（分出后相应值设为1，否则设为0）

int i, n;

// 创建全部扑克牌

initAllCard(arrCardAll);

// 给玩家一发牌

touchCard(arrCardPlayer1, 17, arrCardAll, arrIsUsed);

// 给玩家二发牌

touchCard(arrCardPlayer2, 17, arrCardAll, arrIsUsed);

// 给玩家三发牌

touchCard(arrCardPlayer3, 17, arrCardAll, arrIsUsed);

// 获取底牌

n = 0;

for(i=0; i<54; ++i)

{

if(0 == arrIsUsed[i])

{

arrCardHand[n++] = arrCardAll[i];

arrIsUsed[i] = 1;

}

}

// 对玩家的牌进行排序

sortCard(arrCardPlayer1, 17);

sortCard(arrCardPlayer2, 17);

sortCard(arrCardPlayer3, 17);

// 打印发牌结果

printf("底牌：\n");

printCard(arrCardHand, 3);

printf("玩家一：\n");

printCard(arrCardPlayer1, 17);

printf("玩家二：\n");

printCard(arrCardPlayer2, 17);

printf("玩家三：\n");

printCard(arrCardPlayer3, 17);

return 0;

}

// 初始化全部扑克牌

void initAllCard(Card \* pCard)

{

int i, n, m;

// 循环创建52张带花色的扑克牌

i = 0;

for(n=0; n<4; ++n)

{

for(m=0; m<13; ++m)

{

pCard[i].color = (CardColor)n;

pCard[i].number = (CardNnumber)m;

i++;

}

}

// 创建小王、大王扑克牌

pCard[i].color = None;

pCard[i].number = CardNumKA;

i++;

pCard[i].color = None;

pCard[i].number = CardNumKB;

}

// 给一名玩家发牌

void touchCard(Card \* pPlayer, int nCount, Card \* pAllCard, int \* pIsUsed)

{

int nRandom; // 记录随机数

int i;

// 添加一个随机数种子

srand(time(0));

for(i=0; i<nCount; ++i)

{

// 在0-53之间取随机数

nRandom = rand()%54;

// 判断扑克牌是否已被分出

if(0 == pIsUsed[nRandom])

{

// 扑克牌未分出则分出这张牌

pPlayer[i] = pAllCard[nRandom];

pIsUsed[nRandom] = 1;

}

else

{

// 扑克牌已被分出

i--;

}

}

}

// 比较两张卡牌的大小（包括牌面值与花色），第一张牌大则返回1，否则返回0

int maxCard(Card \* pCard1, Card \* pCard2)

{

if((\*pCard1).number > (\*pCard2).number)

{

return 1;

}

else if((\*pCard1).number == (\*pCard2).number)

{

if((\*pCard1).color > (\*pCard2).color)

{

return 1;

}

}

return 0;

}

// 对玩家的扑克牌进行排序

void sortCard(Card \* pCard, int nCount)

{

Card temCard;

int nMax;

int i, n;

for(i=0; i<nCount; ++i)

{

nMax = i;

for(n=i; n<nCount; ++n)

{

if(1 == maxCard(pCard+n, pCard+nMax))

{

nMax = n;

}

}

temCard = pCard[i];

pCard[i] = pCard[nMax];

pCard[nMax] = temCard;

}

}

// 打印扑克牌数组的所有纸牌信息

void printCard(Card \* pCard, int nCount)

{

int i;

for(i=0; i<nCount; ++i)

{

if(pCard[i].number < CardNumKA)

{

printf("%s%s ", arrStrColor[pCard[i].color], arrStrNumber[pCard[i].number]);

}

else

{

printf("%s ", arrStrNumber[pCard[i].number]);

}

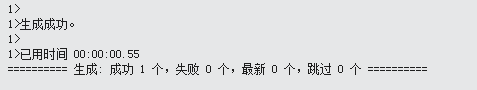
}

printf("\n");

return;

}

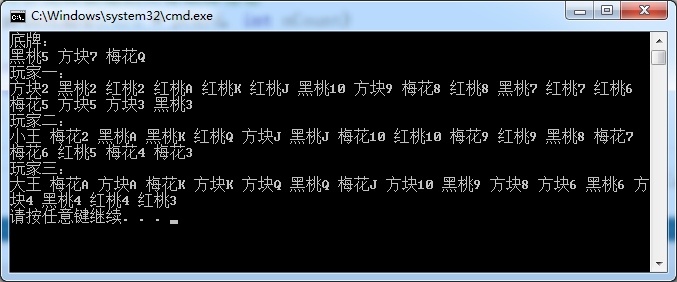
**步骤6：**依次点击“调试”->“开始执行（不调试）”，编译运行程序，没有出现错误的话会在“输出”窗口显示编译成功，并运行程序。



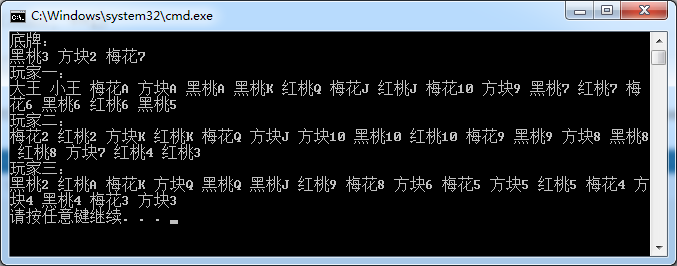
**实验结果/结论**

1. **实验结果**

(1)、



(2)、



2. **实验结论**

* 在函数调用时可以用数组名把数组中第一个结构体的地址传递给被调函数，在被调函数中可以使用数组的方括号符号来访问操作数组中的结构体，但是在将地址作为数组使用时需注意结构体边界问题。

## 11.2 理论题

**注：int类型大小为4字节，char类型大小为1字节，double类型大小为8字节，对齐系数为8。**

### A类

1. **选择题**
2. 下列不属于自定义数据类型的是（）：B
3. 结构体
4. 整型
5. 联合
6. 枚举
7. 下列关于枚举的叙述中不正确的是（ ）： D
8. 枚举变量只能取对应枚举类型的枚举元素表中的元素
9. 可以在定义枚举类型时对枚举元素进行初始化
10. 枚举元素表中的元素有先后次序,可以进行比较
11. 枚举元素的值可以是整数或字符串
12. 以下关于typedef的叙述正确的是（ ）：C
13. 用typedef既定义各种类型名，也可以用来定义变量
14. 用typedef可以增加新类型
15. 用typedef只是将已存在的类型用一个新的名字来代表
16. typedef与 #define的作用是相同的
17. 下列关于结构体叙述不正确的是（ ）：A
18. 结构体成员可以是本身结构体类型
19. 结构体类型的成员名可以与程序中的变量名相同
20. 结构体类型的成员可以是一个结构体变量
21. 可以单独使用结构体变量中的成员，它的作用相当于普通变量
22. 下列可以出现在结构体成员列表中的是（ ）：B
23. 本身结构体类型变量
24. 本身结构体指针类型变量
25. 函数
26. 静态变量
27. 对于结构体类型变量在程序执行期间描述正确的是（ ）：A
28. 所有成员一直驻留在内存中
29. 只有一个成员驻留在内存中
30. 部分成员驻留在内存中
31. 没有成员驻留在内存中
32. 在C语言中，需要把一些属于不同类型的数据作为一个整体来处理时常用（ ）： D
    1. 基本数据类型
    2. 数组
    3. 指针
    4. 结构体类型
33. 在声明一个联合体变量时，系统分配给它的存储空间是（ ）：B
34. 该联合体中第一个成员所需存储空间补齐后大小
35. 该联合体中占用最大存储空间的成员所需存储空间补齐后大小
36. 该联合体中最后一个成员所需的存储空间补齐后大小
37. 该联合体中所有成员所需存储空间的总和补齐后大小
38. 在声明一个结构体变量时，系统分配给它的存储空间是（ ）：D
39. 该结构体变量中第一个成员所需存储空间补齐后大小
40. 该结构体变量中最后一个成员所需存储空间补齐后大小
41. 该结构体变量中占用最大存储空间的成员所需存储空间补齐后大小
42. 该结构体变量中所有成员所需存储空间的总和补齐后大小
43. 以下对C语言中联合体类型数据的叙述正确的是（ ）：B
44. 可以对联合体变量名直接赋值
45. 一个联合体变量中不能同时存放其所有成员
46. 一个联合体变量中可以同时存放其所有成员
47. 联合体类型定义中不能出现结构体类型的成员

### B类

**注：int类型大小为4字节，char类型大小为1字节，double类型大小为8字节，对齐系数为8。**

1. **选择题**
2. 有如下定义，则联合体变量u1在内存中占用的字节数为（ ）：B

union data

{

int a;

char b;

char c[5];

} u1;

1. 5
2. 8
3. 10
4. 12
5. 已知如下结构体类型,则下列结构体变量声明错误的是（ ）：C

typedef struct student

{

char acName[20]；

int nAge；

} Student ;

1. Student s1;
2. struct studetn s1 = {0};
3. student s1 = {"zhangsan", 21};
4. Student s1 = {"zhangsan"} ;
5. 以下枚举类型的定义中正确的是（）： A
6. enum a{one = 9, two = -1, three};
7. enum a = {one, two, three};
8. enum a = {"one", "two", "three"};
9. enum a{"one", "two", "three"};
10. 如下程序的输出结果为（ ）：B

struct point

{

int x ;

int y ;

int z ;

} ;

struct point p1[] = {0,1,2,3,4,5} ;

printf("%d\n", p1[0].x + p1[1].y) ;

1. 3
2. 4
3. 5
4. 6
5. 有如下结构体定义，那么如果要对zhangsan的出生年份进行赋值的话正确的赋值语句是（ ）：D

struct date { int month; int day; int year;};

struct person

{

char name[20];

struct date birthday;

} zhangsan;

1. birthday.year = 1995 ;
2. person.birthday.year = 1995 ;
3. zhangsan.year = 1995 ;
4. zhangsan.birthday.yea r = 1995 ;
5. 有如下联合体定义，那么关于联合体变量初始化正确的是（）：B

union data

{

char \*p;

int n;

};

1. union data u1= "zhangfei";
2. union data u1 = {"zhangfei"};
3. union data u1 = 15;
4. union data u1 = {15};
5. 对如下程序叙述错误的是（ ）：C

struct student

{

char acName[20];

int nAge;

} s1;

1. struct是结构体类型的关键字
2. struct student是用户定义的结构体类型
3. s1是用户定义的结构类型名
4. acName和nAge都是结构成员名

对下面

1. 有如下程序，则以下对结构体变量中成员的引用方式不正确的是（ ）：D

struct book

{

char acName[20] ;

int nPageCount;

} b1, \*pB2;

pB2 = &b1;

1. b1.acName
2. pB2→nPageCount
3. (\*pB2).nPageCount
4. \*p.acName
5. 有如下程序，则此程序的输出结果是（ ）： C

enum team

{

a,

b = 4,

c = a - b,

d = c+ 10

};

printf("%d,%d,%d,%d\n", a, b, c, d);

1. 0,4,5,15
2. 0,4,0,10
3. 0,4,-4,6
4. 0,4,4,14
5. 有如下程序，则此程序的输出结果是（ ）A：

#include <stdio.h>

int main()

{

union

{

short int i[2];

long k;

char c[4];

} r, \*s = &r;

s->i[0] = 0x39;

s->i[1] = 0x38;

printf("%lx\n", s->k);

return 0 ;

}

1. 390038
2. 380039
3. 3939
4. 3838
5. **综合题**

编写一程序能够输入若干本图书的信息并按出版年月排序后进行输出，每本图书包含书名（booktitle）、作者（author）、出版年月（date）、出版社（publishunit）信息。请用一个结构体数组存放图书信息，图书结构体类型已给出，请编写完成程序。

struct Data {

int year;

int month;

int day;

};

struct book

{

char booktitle[50]; // 书名

char author[20]; // 作者

struct Data data; // 出版日期

char publishunit[100]; // 出版社

};

参考程序：

#include <stdio.h>

#include <stdlib.h>

struct Data {

int year;

int month;

int day;

};

struct book

{

char booktitle[50]; // 书名

char author[20]; // 作者

struct Data data; // 出版日期

char publishunit[100]; // 出版社

};

int main(int argc, char \*\* argv)

{

int i, j, n; // 循环变量

struct book \* library; // 存放图书信息的内存地址指针

struct book temp; // 图书临时变量

printf("请输入要处理的图书数量：\n");

scanf("%d", &n);

library = (struct book \*)malloc(n \* sizeof(struct book));

for(i=0; i<n; ++i)

{

printf("请输入第%d本书的信息：\n", i + 1);

printf("书名：");

fflush(stdin);

scanf("%s", &library[i].booktitle);

printf("作者：");

fflush(stdin);

scanf("%s", &library[i].author);

printf("出版年月：");

fflush(stdin);

scanf("%d.%d.%d",

&library[i].data.year,

&library[i].data.month,

&library[i].data.day);

printf("出版社：");

fflush(stdin);

scanf("%s", &library[i].publishunit);

}

for(i=0; i<n-1; ++i)

{

for(j=i+1; j<n; ++j)

{

if((library[i].data.year < library[j].data.year)

|| (library[i].data.year == library[j].data.year

&& library[i].data.month < library[j].data.month)

|| (library[i].data.year == library[j].data.year

&& library[i].data.month == library[j].data.month

&& library[i].data.day < library[j].data.day))

{

temp = library[i];

library[i] = library[j];

library[j] = temp;

}

}

}

printf("\n排序后的图书信息：");

for(i=0; i<n; ++i)

{

printf("\n书名： %s\n 作者： %s\n 出版年月： %d.%d.%d\n 出版社： %s\n",

library[i].booktitle,

library[i].author,

library[i].data.year,

library[i].data.month,

library[i].data.day,

library[i].publishunit);

}

free(library);

return 0;

}

## 本章答案

### A类

**一、选择题**

### B类

**一、选择题**

**二、综合题**