C++实验报告二：有趣的实验

[redacted]

班级：1班

[redacted]

实验1：猜数游戏

实验目的：

1.掌握随机数的使用方法

2.掌握基本的函数的使用方法

3.掌握对一个四位数字的基本操作

4.掌握基本的互动操作

5.掌握对循环的控制方法

实验内容

猜数游戏。由计算机“想”一个四位数，请人猜这个四位数是多少。人输入四位数字后，计算机首先判断这四位数字中有几位是猜对了，并且在对的数字中又有几位位置也是对的，将结果显示出来，给人以提示，请人再猜，直到人猜出计算机所想的四位数是多少为止。

实验方案

```
#include<iostream>
#include<algorithm>
#include<string>
#include<stdlib.h>
#include<random>
using namespace std;
using ull = unsigned long long;
#define rep(i,a,n) for(int i=a;i<=n;i++)
#define frep(i,a,n) for(int i=a;i>=n;i--)
void check(ull x, ull y)
{
    int cnt = 1;
    int ans = 0;
    while (x || y)
    {
        if (x % 10 == y % 10)
        {
            cout << "你猜对了第" << cnt << "位" << '\n';
            ans++;
        }
        x /= 10;
```

```
        y /= 10;
        cnt++;
    }
    if (!ans)
    {
        cout << "太逊了，一位都没猜中" << '\n';
    }
}
int main()
{
    default_random_engine e;
    uniform_int_distribution<int> u(1000,9999);
    cout << "接下来进入猜数的世界吧，少年随我来" << '\n';
    cout << "友好的一些提示如下" << '\n';
    cout << "第一位表示个位" << '\n';
    cout << "第二位表示十位" << '\n';
    cout << "第三位表示百位" << '\n';
    cout << "第四位表示千位" << '\n';
    cout << "请输入你心中的数字并勇往直前罢" << '\n';
    e.seed(time(0));
    ull t = u(e);
    ull x=0;
    cin >> x;
    ull cnt = 1;
    while (x != t)
    {
        if (x > t)
        {
            cout << "少侠输入的数据偏大" << '\n';
            check(x, t);
        }
        else
        {
            cout << "少侠输入的数据偏小" << '\n';
            check(x, t);
        }
        cin >> x;
        cnt++;
    }
    cout << "恭喜少侠，历经" << cnt << "难终成正果" << '\n';
}
```

输出结果

> 接下来进入猜数的世界吧，少年随我来
> 友好的一些提示如下
> 第一位表示个位
> 第二位表示十位
> 第三位表示百位
> 第四位表示千位
> 请输入你心中的数字并勇往直前罢
> 1000
> 少侠输入的数据偏小
> 你猜对了第4位

```
1500
少侠输入的数据偏小
你猜对了第4位
1600
少侠输入的数据偏小
你猜对了第4位
1900
少侠输入的数据偏大
你猜对了第4位
1800
少侠输入的数据偏小
你猜对了第3位
你猜对了第4位
1850
少侠输入的数据偏小
你猜对了第3位
你猜对了第4位
1890
少侠输入的数据偏小
你猜对了第2位
你猜对了第3位
你猜对了第4位
1895
少侠输入的数据偏小
你猜对了第2位
你猜对了第3位
你猜对了第4位
1896
少侠输入的数据偏小
你猜对了第2位
你猜对了第3位
你猜对了第4位
1897
少侠输入的数据偏小
你猜对了第2位
你猜对了第3位
你猜对了第4位
1898
恭喜少侠，历经11难终成正果
```

实验心得

> 经过该实验，我对数字的取法有了更深入的理解，也同时也对一些格式化输出，和用户的使用体验有了进一步的了解。并且，如果是游戏，将是会需要一些提示和鼓励的，在这方面我还略有不足。

实验二：自动发牌

实验目的

> 1.掌握随机数的进阶用法
>
> 2,掌握基本的哈希用法
>
> 3.掌握基本的数组使用方法

实验内容

> 自动发牌。一副扑克有52张牌，打桥牌时应将牌分给四个人。请设计一个程序完成自动发牌的工作。要求：黑桃用S(Spaces)表示；红桃用H(Hearts)表示；方块用D(Diamonds)表示；梅花用C(Clubs)表示。按照打桥牌的规定，每人应当有13张牌。在人工发牌时，先进行洗牌.....

实验方案

```
#include<iostream>
#include<algorithm>
#include<string>
#include<stdlib.h>
#include<random>
#define rep(i,a,n) for(int i=a;i<=n;i++)
#define frep(i,a,n) for(int i=a;i>=n;i--)
using namespace std;
int cnt = 0;
int a[53];
int a1[14];
int a1_cnt;
int a2[14];
int a2_cnt;
int a3[14];
int a3_cnt;
int a4[14];
int a4_cnt;
void print(int x[],int n,int cnt)
{
    rep(i, 1, n)
    {
        cout << "第"<<cnt<<"个人抽到的第" << i << "张牌是" << '\n';
        int t = x[i] % 4;
        int r = x[i] % 13;
        if (t==1)
        {
            cout << "黑桃";
        }
        else if (t == 2)
        {
            cout << "红桃" ;
        }
        else if (t == 3)
        {
            cout << "方块" ;
```

```
        }
        else {
            cout << "梅花" ;
        }

        if (r == 0)
        {
            cout << "K";
        }
        else if (r == 12)
        {
            cout << "Q";
        }
        else if (r == 11)
        {
            cout << "J";
        }
        else {
            cout << r;
        }
        cout << "\n";
    }
}
int main()
{
    default_random_engine e;
    uniform_int_distribution<int> u(1, 52);
    while (cnt != 52)
    {
        int t = u(e);
        if (!a[t])
        {
            cnt++;
            a[t] = 1;
            if (cnt <= 13)
            {
                a1[++a1_cnt] = t;
            }
            else if (cnt > 13 && cnt <= 26)
            {
                a2[++a2_cnt] = t;
            }
            else if (cnt > 26 && cnt <= 39)
            {
                a3[++a3_cnt] = t;
            }
            else if (cnt > 39 && cnt <= 52)
            {
                a4[++a4_cnt] = t;
            }
        }
    }
    print(a1, 13, 1);
    cout << '\n';
    print(a2, 13, 2);
    cout << '\n';
```

```
    print(a3, 13, 3);
    cout << '\n';
    print(a4, 13, 4);
    cout << '\n';
}
```

实验结果输出

第1个人抽到的第1张牌是
方块4
第1个人抽到的第2张牌是
梅花8
第1个人抽到的第3张牌是
梅花9
第1个人抽到的第4张牌是
梅花5
第1个人抽到的第5张牌是
方块7
第1个人抽到的第6张牌是
方块Q
第1个人抽到的第7张牌是
梅花Q
第1个人抽到的第8张牌是
黑桃7
第1个人抽到的第9张牌是
黑桃4
第1个人抽到的第10张牌是
红桃6
第1个人抽到的第11张牌是
黑桃3
第1个人抽到的第12张牌是
方块2
第1个人抽到的第13张牌是
红桃10

第2个人抽到的第1张牌是
梅花K
第2个人抽到的第2张牌是
红桃J
第2个人抽到的第3张牌是
黑桃9
第2个人抽到的第4张牌是
红桃Q
第2个人抽到的第5张牌是
红桃K
第2个人抽到的第6张牌是
红桃3
第2个人抽到的第7张牌是
梅花3
第2个人抽到的第8张牌是
黑桃1

第2个人抽到的第9张牌是
红桃9
第2个人抽到的第10张牌是
红桃8
第2个人抽到的第11张牌是
红桃7
第2个人抽到的第12张牌是
方块1
第2个人抽到的第13张牌是
方块9

第3个人抽到的第1张牌是
红桃2
第3个人抽到的第2张牌是
方块6
第3个人抽到的第3张牌是
黑桃6
第3个人抽到的第4张牌是
黑桃10
第3个人抽到的第5张牌是
梅花10
第3个人抽到的第6张牌是
黑桃8
第3个人抽到的第7张牌是
梅花1
第3个人抽到的第8张牌是
方块K
第3个人抽到的第9张牌是
黑桃Q
第3个人抽到的第10张牌是
黑桃J
第3个人抽到的第11张牌是
方块3
第3个人抽到的第12张牌是
方块10
第3个人抽到的第13张牌是
梅花7

第4个人抽到的第1张牌是
方块J
第4个人抽到的第2张牌是
梅花J
第4个人抽到的第3张牌是
黑桃2
第4个人抽到的第4张牌是
红桃4
第4个人抽到的第5张牌是
红桃5
第4个人抽到的第6张牌是
方块5
第4个人抽到的第7张牌是

红桃1
第4个人抽到的第8张牌是
方块8
第4个人抽到的第9张牌是
黑桃K
第4个人抽到的第10张牌是
梅花4
第4个人抽到的第11张牌是
梅花2
第4个人抽到的第12张牌是
梅花6
第4个人抽到的第13张牌是
黑桃5

实验心得

> 通过本次实验，我更加了解了取余%对于分类这一思想的妙用，同时对于随机数存储的方法也有了更深入的理解，还有就是对于输入方面的美观度也进行了调整，保证结果的输入清晰易懂。

实验三：数字移动

实验目的

> 1.掌握对map和pair的使用（或者对一维数组的使用，两种方法都可以）
>
> 2.掌握随机数的生成方法
>
> 3.掌握结构体的使用方法
>
> 4.掌握基本的交换方法
>
> 5。掌握函数的调用

实验内容

> 【1】 在图中的九个点上,空出中间的点,其余的点上任意填入数字1到8;1的位置固定不动,然后移动其余的数字,使1到8顺时针从小到大排列.移动的规律是:只能将数字沿线移向空白的点。请编程显示数字移动过程。

实验方案

```
#include<iostream>
#include<algorithm>
#include<string>
#include<stdlib.h>
#include<random>
#include<map>
#include<time.h>
#define rep(i,a,n) for(int i=a;i<=n;i++)
#define frep(i,a,n) for(int i=a;i>=n;i--)
using namespace std;
int a[9];
int  b[9];
int cnt;
```

```
int shu[4][4];
//记录每一个点的位置，意思是point[2].x就是2所在的矩阵的横坐标，y就是纵坐标
struct dian
{
    int x;
    int y;
}point[9];
//打印函数
void print()
{
    rep(i, 1, 3)
    {
        rep(j, 1, 3)
        {
            cout << shu[i][j] << ' ';
        }
        cout << "\n";
    }
    cout<<"--------------------------------"<<'\n';
}
map<pair<int, int>, pair<int, int> >q;
map<pair<int, int>, pair<int, int> >p;
void myswap(int i, int j, int k,int l)
{

    int x = point[shu[i][j]].x;
    int y = point[shu[i][j]].y;
    //首先shu[i][j]是矩阵的数字，如果是shu[i][j]=1相当于我们记录下1的坐标
    point[shu[i][j]].x = point[shu[k][l]].x;
    point[shu[i][j]].y = point[shu[k][l]].y;
    //这是交换点的坐标
    point[shu[k][l]].x = x;
    point[shu[k][l]].y = y;
    //也是交换
       int temp = shu[i][j];
    shu[i][j] = shu[k][l];
    shu[k][l] = temp;
    //这三行是矩阵值的交换了 ，例如矩阵shu[i][[j]和shu[k][l]交换
}
int main()
{
   // default_random_engine e;
    //uniform_int_distribution<int> u(1, 8);
    //生成随机数种子
    srand(time(0));
    while (cnt < 8)
    {
        int t=rand()%8+1;
        if (!a[t])
        {
            cnt++;
            //记录下我们收集了1-8的数字收集了多少个
            a[t]++;
            //如果出现了标记一下
            b[cnt] = t;
            //弄一个数组记录下数字出现的先后，达到随机的目的
```

```
        }
    }
    //矩阵的初始化
    shu[1][1] = b[1];
    shu[2][1] = b[2];
    shu[3][1] = b[3];
    shu[3][2] = b[4];
    shu[3][3] = b[5];
    shu[2][3] = b[6];
    shu[1][3] = b[7];
    shu[1][2] = b[8];
    //初始矩阵的打印
    print();
    rep(i, 1, 3)
    {
        rep(j, 1, 3)
        {
            //记录下矩阵的每一个值的横纵坐标
            point[shu[i][j]].x = i;
            point[shu[i][j]].y = j;
        }
    }
//因为我们中间是需要一个0的，我们把0这个数字的横纵坐标也初始化一下
    point[0].x = 2;
    point[0].y = 2;
    int cnt_ = 0;
    //可以理解为这是顺时针的一个查询操作，
    //我们想象一个3*3的矩阵，顺时针方向，
    //是不是1，1指向1，2，
    //（1，2）再指向1，3
q[make_pair(1, 1)] = make_pair(1, 2);
q[make_pair(1, 2)] = make_pair(1, 3);
q[make_pair(1, 3)] = make_pair(2, 3);
q[make_pair(2, 3)] = make_pair(3, 3);
q[make_pair(3,3)] = make_pair(3, 2);
q[make_pair(3, 2)] = make_pair(3,1 );
q[make_pair(3, 1)] = make_pair(2, 1);
q[make_pair(2, 1)] = make_pair(1, 1);

    //可以理解为这是逆时针的一个查询操作，
    //我们想象一个3*3的矩阵，顺时针方向，
    //是不是1，1指向2，1，
    //（2，1）再指向3，1
p[{1, 1}] = { 2, 1 };
p[make_pair(2, 1)] = make_pair(3, 1);
p[make_pair(3, 1)] = make_pair(3, 2);
p[make_pair(3, 2)] = make_pair(3, 3);
p[make_pair(3, 3)] = make_pair(2, 3);
p[make_pair(2,3)] = make_pair(1, 3);
p[make_pair(1, 3)] = make_pair(1, 2);
p[make_pair(1, 2)] = make_pair(1, 1);

//接下来是主操作部分
rep(i, 1, 7)
{
    //如果当前的数字i的顺时针方向的下一个数字不是i+1,好比1的下一个数字不是2
```

```
    if (shu[q[make_pair(point[i].x, point[i].y)].first][q[make_pair(point[i].x,
point[i].y)].second] != i + 1)
    {
        int x = point[i].x;
        int y = point[i].y;
        //记录下来这个不能换的数字，也就是当前的数字，好比是1，如果当前i=2，就是2,因为通过i=1
的操作2已经有序了，所以不用继续是1
        myswap(2, 2, point[i + 1].x, point[i + 1].y);
        //先把要换的数字和2，2位置的0交换位置
        print();
        //打印操作
        int xnow = point[0].x;
        int ynow = point[0].y;
        //记录下0的坐标，注意0不在2，2位置了，而是在我们要交换的那个数字的位置
        int x11 = p[make_pair(xnow, ynow)].first;
        int y11 = p[make_pair(xnow, ynow)].second;
        //这是0的逆时针方向的数字的位置
        while (x11 != x && y11 != y)
        {
            myswap(x11, y11, xnow, ynow);
            xnow = point[0].x;
            ynow = point[0].y;
            x11 = p[make_pair(xnow, ynow)].first;
            y11 = p[make_pair(xnow, ynow)].second;
            print();
        }
        //上面是类似于冒泡排序，就不断交换0和逆时针方向的数字的过程，直到那个数字是有序的为止
        myswap(point[0].x, point[0].y, q[make_pair(point[i].x,
point[i].y)].first, q[make_pair(point[i].x, point[i].y)].second);
        print();
        myswap(2, 2, point[0].x, point[0].y);
        //最后取出那个在2，2位置的数字，我们一开始把他扔进去了，现在把0弄回去，把他弄回来
        print();
       // 最后打印
    }
}
}
```

实验结果输出

```
345
601
```

287

```
345
621
```

087

345
621

807

345
621

870

345
601

872

045
631

872

645
031

872

645
731

802

645
701

832

605
741

832

065
741

832

765
041

8 3 2

7 6 5
8 4 1

0 3 2

7 6 5
8 0 1

4 3 2

7 6 0
8 5 1

4 3 2

7 0 6
8 5 1

4 3 2

7 8 6
0 5 1

4 3 2

7 8 6
5 0 1

4 3 2

7 8 0
5 6 1

4 3 2

7 0 8
5 6 1

4 3 2

0 7 8
5 6 1

4 3 2

6 7 8
5 0 1

> 4 3 2

实验心得

> 通过这次实验，我对map存储数据有了更深入的了解，之前对map存储键值对的方式还不太清晰，现在则有了大致的概念。同时，我对结构体存储点的方法有了。并且，交换点的坐标的一些考虑也是需要考虑的。

实验四：黑白棋子交换

实验目的

> 1.掌握对题目要求的分析
>
> 2.熟练掌握分类讨论的能力
>
> 3.熟练掌握对交换函数的书写

实验内容

> 【1】 游戏的目的是用最少的步数将上图中白子和黑子的位置进行交换：游戏的规则是：(1)一次只能移动一个棋子； (2)棋子可以向空格中移动，也可以跳过一个对方的棋子进入空格，但不能向后跳，也不能跳过两个子。请用计算机实现上述游戏。例如：
>
> 有三个白子和三个黑子如下图布置：
>
> ○ ○ ○ ● ● ●
>
> 游戏的目的是用最少的步数将上图中白子和黑子的位置进行交换：
>
> ● ● ● ○ ○ ○
>
> 黑白子可用不同的符号来表示。

实验方案

```
//任务四：黑白棋子转化
#include<iostream>
#include<algorithm>
#include<string>
#include<stdlib.h>
#include<random>
#include<map>
#include<time.h>
#define rep(i,a,n) for(int i=a;i<=n;i++)
#define frep(i,a,n) for(int i=a;i>=n;i--)
using namespace std;
void print(int a[])
{
    int i;

    printf(" ");
    for (i = 0; i <= 6; i++)
        printf("  %c", a[i] == 1 ? '*' : (a[i] == 2 ? '@' : ' '));
    printf("\n.............................\n");
```

```
}
void myswap(int* n, int* m)
{
    int term;
    term = *n;
    *n = *m;
    *m = term;
}
int t[7] = { 1,1,1,0,2,2,2 };
int main()
{

    int i, flag;
    print(t);
    while (t[0] + t[1] + t[2] != 6 || t[4] + t[5] + t[6] != 3)
    {
        flag = 1;
        for (i = 0; flag && i < 5; i++)
            if (t[i] == 1 && t[i + 1] == 2 && t[i + 2] == 0)
            {
                myswap(&t[i], &t[i + 2]);
                print(t);
                flag = 0;
            }
        for (i = 0; flag && i < 5; i++)
            if (t[i] == 0 && t[i + 1] == 1 && t[i + 2] == 2)
            {
               myswap(&t[i], &t[i + 2]);
               print(t);
               flag = 0;
            }
        for (i = 0; flag && i < 6; i++)
            if (t[i] == 1 && t[i + 1] == 0 && (i == 0 || t[i - 1] != t[i + 2]))
            {
                myswap(&t[i], &t[i + 1]);
                print(t);
                flag = 0;
            }
        for (i = 0; flag && i < 6; i++)
            if (t[i] == 0 && t[i + 1] == 2 && (i == 5 || t[i - 1] != t[i + 2]))
            {
                myswap(&t[i], &t[i + 1]);
                print(t);
                flag = 0;
            }
    }
}
```

实验结果输出

```
   *  *  *     @  @  @
............................
   *  *     *  @  @  @
............................
   *  *  @  *     @  @
```

```
.............................
  *  *  @  *  @     @
.............................
  *  *  @     @  *  @
.............................
  *     @  *  @  *  @
.............................
     *  @  *  @  *  @
.............................
  @  *     *  @  *  @
.............................
  @  *  @  *     *  @
.............................
  @  *  @  *  @  *
.............................
  @  *  @  *  @     *
.............................
  @  *  @     @  *  *
.............................
  @     @  *  @  *  *
.............................
  @  @     *  @  *  *
.............................
  @  @  @  *     *  *
.............................
  @  @  @     *  *  *
.............................
```

实验心得

> 通过该实验，我掌握了如何分类讨论，纸上模拟一个问题的过程，并且对各种特判进行相应的讨论和分析，同时也掌握了一定的建模能力，把字符串换种整数分析，最后再换回来。

实验五：洛谷P 1598

实验目的

> 1.掌握对字符串的读入方法
>
> 2.对字符串的一些思维变换
>
> 3.对字符串的格式化输出

实验内容

> 写一个程序从输入文件中去读取四行大写字母（全都是大写的，每行不超过 100100 个字符），然后用柱状图输出每个字符在输入文件中出现的次数。严格地按照输出样例来安排你的输出格式。（洛谷P 1598 ）

实验输入样例

```
THE QUICK BROWN FOX JUMPED OVER THE LAZY DOG.
THIS IS AN EXAMPLE TO TEST FOR YOUR
HISTOGRAM PROGRAM.
HELLO!
```

实验方案

```
#include<iostream>
#include<algorithm>
#include<string>
#include<stdlib.h>
#include<random>
#define rep(i,a,n) for(int i=a;i<=n;i++)
#define frep(i,a,n) for(int i=a;i>=n;i--)
using namespace std;
int ff[26];
int  maxn = 0;
int n;
int main()
{
    string a;
  rep(i,1,4)
  {
        getline(cin, a);
        n = a.length();
        rep(j, 0, n-1)
        {
            if (a[j] >= 'A' && a[j] <= 'Z')
            {
                ff[a[j] - 'A']++;
            }
        }
    }
  rep(i, 0, 26)
  {
      maxn = max(maxn, ff[i]);
  }
  frep(i, maxn, 1)
  {
      rep(j, 0, 25)
      {
          if (ff[j] >= i)
          {
              cout << "* ";
          }
          else
          {
              printf("  ");
          }
      }
      printf("\n");
```

```
  }
  rep(i,0,25)
    {
        cout << char(i + 'A')<<' ';
    }
}
```

实验输出

```
THE QUICK BROWN FOX JUMPED OVER THE LAZY DOG.
THIS IS AN EXAMPLE TO TEST FOR YOUR
HISTOGRAM PROGRAM.
HELLO!
                              *
                              *
         *                    *
         *                    *     *   *
         *                    *     *   *
*        *     *              *     *   *
*        *     * *     * *    *     * * *
*        *   * * *     * *    * *   * * * *
*     * * * * * *     * * * * *   * * * *     * *
* * * * * * * * * * * * * * * * * * * * * * * * * *
A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
```

实验心得

> 通过本次实验，我对观看样例的输出有了一定的感悟，并且结合样例输出调整了方法，并且对字符的输出也需要一定的强制类型，以上就是心得。

实验日期

> 11月17日