《数据结构》第二次作业

线性表基本算法

# 一、查找算法

## 顺序表的查找

随机生成100个1~100之间的整数，再随机生成一个1~100之间的整数*a*，查找a是否在前100个整数中，如果不在，返回“不在数列中”；否则，返回数据位置。

要求：

1. 显示产生的100个数据，按如下形式显示：0:21, 1:10，… 99:5；
2. 显示要查找的数据a；
3. 显示查找结果（位置）及查找次数。

程序代码：

#pragma GCC optimize(1)

#pragma GCC optimize(2)

#pragma GCC optimize(3."Ofast","inline")

#include<bits/stdc++.h>

#include<regex>

using namespace std;

#define \_for(i,a,b) for(i=(a);i<(b);i++)

#define \_foreq(i,a,b) for(i=(a);i<=(b);i++)

int main(){

    int n[100];

    int i;

    srand(time(0));

    \_for(i,0,99){

        n[i] = rand()%100+1;

        cout<<i<<":"<<n[i]<<", ";

    }

    n[i] = rand()%100+1;

    cout<<i<<":"<<n[i]<<"\n";

    int a = rand()%100+1;

    cout<<"a:"<<a<<"\n";

    \_for(i,0,100){

        if(a == n[i]){

            cout<<"Result index: "<<i<<" "<<"find times: "<<i+1<<endl;

            break;

        }

    }

    if(i == 100){

        cout<<"Not in the array"<<endl;

    }

    return 0;

}

运行结果：

第一次：

0:39, 1:55, 2:45, 3:51, 4:34, 5:26, 6:2, 7:98, 8:86, 9:51, 10:66, 11:92, 12:19, 13:14, 14:85, 15:80, 16:71, 17:98, 18:62, 19:34, 20:96, 21:48, 22:46, 23:41, 24:98, 25:39, 26:51, 27:65, 28:66, 29:25, 30:97, 31:98, 32:50, 33:66, 34:10, 35:57, 36:7, 37:16, 38:63, 39:42, 40:77, 41:11, 42:57, 43:15, 44:94, 45:90, 46:29, 47:8, 48:2, 49:76, 50:100, 51:96, 52:85, 53:12, 54:90, 55:67, 56:94, 57:55, 58:93, 59:44, 60:90, 61:28, 62:24, 63:53, 64:4, 65:86, 66:20, 67:36, 68:27, 69:52, 70:79, 71:12, 72:32, 73:69, 74:15, 75:49, 76:79, 77:41, 78:93, 79:92, 80:84, 81:8, 82:95, 83:51, 84:2, 85:73, 86:75, 87:80, 88:26, 89:50, 90:41, 91:39, 92:20, 93:50, 94:19, 95:80, 96:44, 97:87, 98:87, 99:13

a:15

Result index: 43 find times: 44

第二次：

0:76, 1:36, 2:77, 3:2, 4:62, 5:9, 6:53, 7:25, 8:45, 9:76, 10:54, 11:42, 12:48, 13:32, 14:52, 15:84, 16:47, 17:25, 18:16, 19:30, 20:95, 21:55, 22:9, 23:42, 24:31, 25:88, 26:6, 27:15, 28:86, 29:30, 30:64, 31:5, 32:30, 33:24, 34:10, 35:67, 36:44, 37:3, 38:42, 39:61, 40:13, 41:56, 42:45, 43:96, 44:25, 45:4, 46:98, 47:46, 48:91, 49:46, 50:3, 51:15, 52:16, 53:76, 54:89, 55:20, 56:10, 57:41, 58:60, 59:81, 60:19, 61:27, 62:34, 63:54, 64:22, 65:68, 66:48, 67:85, 68:83, 69:45, 70:74, 71:75, 72:74, 73:33, 74:20, 75:84, 76:73, 77:79, 78:32, 79:68, 80:65, 81:79, 82:7, 83:79, 84:57, 85:78, 86:94, 87:51, 88:95, 89:80, 90:54, 91:45, 92:42, 93:94, 94:63, 95:34, 96:45, 97:88, 98:100, 99:30

a:80

Result index: 89 find times: 90

第三次：

0:81, 1:5, 2:6, 3:64, 4:49, 5:60, 6:47, 7:56, 8:28, 9:46, 10:21, 11:52, 12:96, 13:26, 14:5, 15:47, 16:55, 17:8, 18:63, 19:56, 20:43, 21:57, 22:32, 23:33, 24:74, 25:89, 26:52, 27:49, 28:81, 29:92, 30:91, 31:53, 32:71, 33:36, 34:48, 35:54, 36:92, 37:18, 38:31, 39:14, 40:43, 41:46, 42:67, 43:80, 44:53, 45:48, 46:39, 47:9, 48:11, 49:92, 50:63, 51:33, 52:22, 53:34, 54:29, 55:73, 56:44, 57:6, 58:61, 59:42, 60:66, 61:34, 62:56, 63:25, 64:4, 65:6, 66:72, 67:32, 68:16, 69:11, 70:52, 71:58, 72:51, 73:35, 74:53, 75:7, 76:48, 77:96, 78:14, 79:30, 80:54, 81:43, 82:72, 83:74, 84:56, 85:75, 86:57, 87:50, 88:91, 89:90, 90:61, 91:32, 92:4, 93:18, 94:58, 95:21, 96:43, 97:80, 98:48, 99:75

a:79

Not in the array

## 有序表的折半查找算法

随机生成100个单调增的整数，在生成一个在最大值和最小值之间的整数a，查找a是否在前100个整数中，如果不在，返回“不在数列中”；否则，返回数据位置。

要求：

1. 用折半查找法查找数据
2. 显示产生的100个数据，按如下形式显示：0:21, 1:10，… 99:5；
3. 显示要查找的数据a；
4. 显示查找结果（位置）及查找次数。

程序代码：

#pragma GCC optimize(1)

#pragma GCC optimize(2)

#pragma GCC optimize(3."Ofast","inline")

#include<bits/stdc++.h>

#include<regex>

using namespace std;

#define \_for(i,a,b) for(i=(a);i<(b);i++)

#define \_foreq(i,a,b) for(i=(a);i<=(b);i++)

#define \_for\_m(i,a,b) for(i=(a);i>(b);i--)

#define \_for\_meq(i,a,b) for(i=(a);i>=(b);i--)

int bisearch(int a[], int t, int left, int right, int \*p){

    int m;

    while(left <= right){

        (\*p)++;

        m = (left + right)/2;

        if(t == a[m]){

            return m;

        }

        else if(t < a[m]) right = m - 1;

        else left = m + 1;

    }

    return -1;

}

int main(){

    int n[100],temp;

    int i,j,k;

    srand(time(0));

    \_for(i,0,100){

        temp = rand()%300+1;

        \_for(j,0,i){

            if(temp < n[j]){

                \_for\_meq(k,i-1,j){

                    n[k+1] = n[k];

                }

                break;

            }

        }

        n[j] = temp;

    }

    \_for(i,0,99){

        cout<<i<<":"<<n[i]<<", ";

    }

    cout<<i<<":"<<n[i]<<"\n";

    int a = rand()%300+1;

    cout<<"\na:"<<a<<"\n";

    int cnt = 0;

    int index = bisearch(n, a, 0, 99, &cnt);

    if(index == -1){

        cout<<"Not in the array"<<endl;

    }

    else cout<<"Result index: "<<index<<" "<<"find times: "<<cnt<<endl;

    return 0;

}

运行结果：

第一次：

0:5, 1:5, 2:6, 3:7, 4:8, 5:8, 6:8, 7:10, 8:12, 9:16, 10:17, 11:21, 12:27, 13:29, 14:31, 15:33, 16:33, 17:34, 18:37, 19:38, 20:40, 21:43, 22:47, 23:50, 24:51, 25:51, 26:51, 27:51, 28:52, 29:54, 30:68, 31:70, 32:74, 33:78, 34:79, 35:82, 36:84, 37:95, 38:99, 39:105, 40:108, 41:111, 42:113, 43:114, 44:115, 45:118, 46:124, 47:128, 48:131, 49:131, 50:136, 51:137, 52:138, 53:145, 54:147, 55:148, 56:149, 57:153, 58:156, 59:157, 60:162, 61:165, 62:166, 63:169, 64:177, 65:183, 66:186, 67:195, 68:199, 69:201, 70:213, 71:217, 72:224, 73:229, 74:229, 75:231, 76:233, 77:235, 78:240, 79:240, 80:243, 81:247, 82:249, 83:249, 84:256, 85:257, 86:261, 87:262, 88:264, 89:265, 90:267, 91:271, 92:272, 93:281, 94:284, 95:289, 96:293, 97:295, 98:299, 99:300

a:259

Not in the array

第二次：

0:9, 1:9, 2:11, 3:12, 4:17, 5:18, 6:20, 7:21, 8:22, 9:28, 10:32, 11:37, 12:52, 13:54, 14:65, 15:66, 16:66, 17:66, 18:69, 19:76, 20:78, 21:80, 22:82, 23:83, 24:89, 25:90, 26:96, 27:97, 28:99, 29:99, 30:99, 31:101, 32:101, 33:106, 34:108, 35:114, 36:115, 37:121, 38:130, 39:134, 40:140, 41:142, 42:148, 43:152, 44:153, 45:160, 46:162, 47:164, 48:169, 49:170, 50:174, 51:174, 52:176, 53:179, 54:180, 55:185, 56:189, 57:190, 58:190, 59:192, 60:201, 61:201, 62:202, 63:203, 64:207, 65:212, 66:216, 67:216, 68:218, 69:218, 70:220, 71:223, 72:223, 73:224, 74:227, 75:230, 76:233, 77:235, 78:237, 79:238, 80:241, 81:242, 82:242, 83:246, 84:249, 85:255, 86:260, 87:260, 88:262, 89:263, 90:263, 91:268, 92:274, 93:279, 94:281, 95:282, 96:292, 97:295, 98:295, 99:296

a:90

Result index: 25 find times: 6

第三次：

0:6, 1:10, 2:18, 3:22, 4:27, 5:29, 6:30, 7:30, 8:34, 9:36, 10:36, 11:46, 12:48, 13:51, 14:54, 15:57, 16:59, 17:59, 18:61, 19:64, 20:64, 21:65, 22:70, 23:71, 24:77, 25:77, 26:78, 27:79, 28:84, 29:87, 30:90, 31:93, 32:94, 33:95, 34:96, 35:106, 36:108, 37:110, 38:113, 39:115, 40:117, 41:117, 42:118, 43:120, 44:121, 45:134, 46:136, 47:143, 48:146, 49:148, 50:154, 51:155, 52:156, 53:157, 54:161, 55:166, 56:170, 57:173, 58:179, 59:184, 60:190, 61:191, 62:191, 63:195, 64:202, 65:204, 66:204, 67:207, 68:209, 69:212, 70:214, 71:218, 72:221, 73:226, 74:227, 75:229, 76:230, 77:234, 78:235, 79:237, 80:239, 81:242, 82:249, 83:252, 84:253, 85:256, 86:262, 87:269, 88:271, 89:271, 90:273, 91:274, 92:275, 93:281, 94:283, 95:287, 96:288, 97:295, 98:296, 99:299

a:148

Result index: 49 find times: 1

# 二、有序表的合并

## 将连个有序表合并成一个有序表

随机生成两个100个1~100之间的整数序列a和b，对这两个序列进行排序；再将这两个有序序列进行合并成整数序列c，使得c也是有序的。

要求：

1. 显示原始序列a和b；
2. 显示a和b排序后的序列；
3. 显示合并和的有序序列c。

程序代码：

#pragma GCC optimize(1)

#pragma GCC optimize(2)

#pragma GCC optimize(3."Ofast","inline")

#include<bits/stdc++.h>

#include<regex>

using namespace std;

#define \_for(i,a,b) for(i=(a);i<(b);i++)

#define \_foreq(i,a,b) for(i=(a);i<=(b);i++)

#define \_for\_m(i,a,b) for(i=(a);i>(b);i--)

#define \_for\_meq(i,a,b) for(i=(a);i>=(b);i--)

int main(){

    ios::sync\_with\_stdio(false);

    int a[100],b[100],c[200];

    int i,j,k;

    int temp;

    srand(time(0));

    \_for(i,0,100){

        temp = rand()%300+1;

        \_for(j,0,i){

            if(temp < a[j]){

                \_for\_meq(k,i-1,j){

                    a[k+1] = a[k];

                }

                break;

            }

        }

        a[j] = temp;

        temp = rand()%300+1;

        \_for(j,0,i){

            if(temp < b[j]){

                \_for\_meq(k,i-1,j){

                    b[k+1] = b[k];

                }

                break;

            }

        }

        b[j] = temp;

    }

    cout<<"a"<<"\n";

    \_for(i,0,99){

        cout<<i<<":"<<a[i]<<", ";

    }

    cout<<i<<":"<<a[i]<<"\n";

    cout<<"b"<<"\n";

    \_for(i,0,99){

        cout<<i<<":"<<b[i]<<", ";

    }

    cout<<i<<":"<<b[i]<<"\n";

    i = 0; j = 0; k = 0;

    while(i < 100 && j < 100){

        if(a[i] < b[j]){

            c[k] = a[i];

            i++;

        }

        else{

            c[k] = b[j];

            j++;

        }

        k++;

    }

    if(i < 100){

        \_for(i,i,100){

            c[k] = a[i];

            k++;

        }

    }

    else{

        \_for(j,j,100){

            c[k] = b[j];

            k++;

        }

    }

    cout<<"c"<<"\n";

    \_for(k,0,199){

        cout<<k<<":"<<c[k]<<", ";

    }

    cout<<k<<":"<<c[k]<<"\n";

    return 0;

}

运行结果：

第一次：

a

0:1, 1:9, 2:13, 3:24, 4:25, 5:27, 6:27, 7:30, 8:32, 9:37, 10:37, 11:43, 12:43, 13:45, 14:51, 15:52, 16:54, 17:54, 18:57, 19:66, 20:67, 21:77, 22:78, 23:79, 24:83, 25:83, 26:85, 27:86, 28:102, 29:106, 30:108, 31:110, 32:110, 33:111, 34:111, 35:112, 36:113, 37:114, 38:117, 39:118, 40:120, 41:127, 42:127, 43:129, 44:132, 45:140, 46:141, 47:151, 48:151, 49:151, 50:161, 51:161, 52:166, 53:167, 54:167, 55:170, 56:171, 57:173, 58:174, 59:178, 60:181, 61:181, 62:181, 63:190, 64:196, 65:200, 66:202, 67:202, 68:206, 69:208, 70:209, 71:210, 72:212, 73:216, 74:217, 75:217, 76:224, 77:224, 78:225, 79:229, 80:232, 81:237, 82:238, 83:243, 84:246, 85:249, 86:251, 87:253, 88:259, 89:259, 90:267, 91:270, 92:270, 93:281, 94:282, 95:282, 96:292, 97:295, 98:296, 99:299

b

0:6, 1:7, 2:7, 3:15, 4:16, 5:16, 6:17, 7:17, 8:17, 9:20, 10:23, 11:24, 12:32, 13:32, 14:35, 15:37, 16:45, 17:55, 18:57, 19:59, 20:61, 21:63, 22:64, 23:65, 24:67, 25:68, 26:70, 27:74, 28:75, 29:77, 30:82, 31:88, 32:91, 33:94, 34:95, 35:97, 36:101, 37:101, 38:107, 39:108, 40:110, 41:114, 42:121, 43:129, 44:131, 45:131, 46:140, 47:142, 48:145, 49:146, 50:147, 51:155, 52:158, 53:159, 54:168, 55:171, 56:177, 57:177, 58:178, 59:182, 60:183, 61:184, 62:184, 63:190, 64:191, 65:192, 66:195, 67:202, 68:202, 69:203, 70:208, 71:209, 72:210, 73:210, 74:222, 75:223, 76:233, 77:235, 78:251, 79:252, 80:253, 81:254, 82:260, 83:261, 84:261, 85:262, 86:264, 87:267, 88:267, 89:269, 90:271, 91:272, 92:275, 93:281, 94:287, 95:290, 96:293, 97:294, 98:299, 99:300

c

0:1, 1:6, 2:7, 3:7, 4:9, 5:13, 6:15, 7:16, 8:16, 9:17, 10:17, 11:17, 12:20, 13:23, 14:24, 15:24, 16:25, 17:27, 18:27, 19:30, 20:32, 21:32, 22:32, 23:35, 24:37, 25:37, 26:37, 27:43, 28:43, 29:45, 30:45, 31:51, 32:52, 33:54, 34:54, 35:55, 36:57, 37:57, 38:59, 39:61, 40:63, 41:64, 42:65, 43:66, 44:67, 45:67, 46:68, 47:70, 48:74, 49:75, 50:77, 51:77, 52:78, 53:79, 54:82, 55:83, 56:83, 57:85, 58:86, 59:88, 60:91, 61:94, 62:95, 63:97, 64:101, 65:101, 66:102, 67:106, 68:107, 69:108, 70:108, 71:110, 72:110, 73:110, 74:111, 75:111, 76:112, 77:113, 78:114, 79:114, 80:117, 81:118, 82:120, 83:121, 84:127, 85:127, 86:129, 87:129, 88:131, 89:131, 90:132, 91:140, 92:140, 93:141, 94:142, 95:145, 96:146, 97:147, 98:151, 99:151, 100:151, 101:155, 102:158, 103:159, 104:161, 105:161, 106:166, 107:167, 108:167, 109:168, 110:170, 111:171, 112:171, 113:173, 114:174, 115:177, 116:177, 117:178, 118:178, 119:181, 120:181, 121:181, 122:182, 123:183, 124:184, 125:184, 126:190, 127:190, 128:191, 129:192, 130:195, 131:196, 132:200, 133:202, 134:202, 135:202, 136:202, 137:203, 138:206, 139:208, 140:208, 141:209, 142:209, 143:210, 144:210, 145:210, 146:212, 147:216, 148:217, 149:217, 150:222, 151:223, 152:224, 153:224, 154:225, 155:229, 156:232, 157:233, 158:235, 159:237, 160:238, 161:243, 162:246, 163:249, 164:251, 165:251, 166:252, 167:253, 168:253, 169:254, 170:259, 171:259, 172:260, 173:261, 174:261, 175:262, 176:264, 177:267, 178:267, 179:267, 180:269, 181:270, 182:270, 183:271, 184:272, 185:275, 186:281, 187:281, 188:282, 189:282, 190:287, 191:290, 192:292, 193:293, 194:294, 195:295, 196:296, 197:299, 198:299, 199:300

第二次：

a

0:3, 1:3, 2:5, 3:5, 4:6, 5:9, 6:12, 7:16, 8:18, 9:18, 10:19, 11:19, 12:21, 13:23, 14:24, 15:29, 16:31, 17:32, 18:38, 19:45, 20:45, 21:46, 22:49, 23:50, 24:51, 25:52, 26:56, 27:57, 28:62, 29:66, 30:69, 31:72, 32:87, 33:90, 34:93, 35:96, 36:97, 37:97, 38:105, 39:107, 40:122, 41:127, 42:128, 43:130, 44:131, 45:139, 46:143, 47:146, 48:147, 49:152, 50:160, 51:162, 52:163, 53:166, 54:167, 55:170, 56:171, 57:171, 58:173, 59:178, 60:178, 61:182, 62:182, 63:187, 64:189, 65:191, 66:191, 67:192, 68:206, 69:213, 70:214, 71:217, 72:218, 73:225, 74:226, 75:229, 76:238, 77:240, 78:241, 79:245, 80:248, 81:248, 82:248, 83:249, 84:250, 85:251, 86:252, 87:253, 88:255, 89:259, 90:263, 91:266, 92:270, 93:275, 94:277, 95:279, 96:286, 97:294, 98:297, 99:297

b

0:5, 1:8, 2:9, 3:13, 4:14, 5:19, 6:28, 7:32, 8:33, 9:33, 10:41, 11:44, 12:48, 13:54, 14:55, 15:55, 16:56, 17:63, 18:70, 19:70, 20:73, 21:77, 22:77, 23:80, 24:83, 25:85, 26:86, 27:87, 28:88, 29:88, 30:90, 31:103, 32:106, 33:107, 34:111, 35:115, 36:115, 37:118, 38:121, 39:122, 40:124, 41:125, 42:136, 43:137, 44:139, 45:144, 46:149, 47:151, 48:153, 49:156, 50:159, 51:160, 52:168, 53:168, 54:169, 55:170, 56:171, 57:173, 58:174, 59:177, 60:178, 61:178, 62:182, 63:186, 64:188, 65:188, 66:190, 67:192, 68:193, 69:195, 70:195, 71:202, 72:203, 73:204, 74:204, 75:206, 76:212, 77:217, 78:218, 79:225, 80:225, 81:229, 82:229, 83:231, 84:236, 85:238, 86:240, 87:241, 88:243, 89:247, 90:262, 91:265, 92:265, 93:268, 94:270, 95:277, 96:278, 97:291, 98:299, 99:300

c

0:3, 1:3, 2:5, 3:5, 4:5, 5:6, 6:8, 7:9, 8:9, 9:12, 10:13, 11:14, 12:16, 13:18, 14:18, 15:19, 16:19, 17:19, 18:21, 19:23, 20:24, 21:28, 22:29, 23:31, 24:32, 25:32, 26:33, 27:33, 28:38, 29:41, 30:44, 31:45, 32:45, 33:46, 34:48, 35:49, 36:50, 37:51, 38:52, 39:54, 40:55, 41:55, 42:56, 43:56, 44:57, 45:62, 46:63, 47:66, 48:69, 49:70, 50:70, 51:72, 52:73, 53:77, 54:77, 55:80, 56:83, 57:85, 58:86, 59:87, 60:87, 61:88, 62:88, 63:90, 64:90, 65:93, 66:96, 67:97, 68:97, 69:103, 70:105, 71:106, 72:107, 73:107, 74:111, 75:115, 76:115, 77:118, 78:121, 79:122, 80:122, 81:124, 82:125, 83:127, 84:128, 85:130, 86:131, 87:136, 88:137, 89:139, 90:139, 91:143, 92:144, 93:146, 94:147, 95:149, 96:151, 97:152, 98:153, 99:156, 100:159, 101:160, 102:160, 103:162, 104:163, 105:166, 106:167, 107:168, 108:168, 109:169, 110:170, 111:170, 112:171, 113:171, 114:171, 115:173, 116:173, 117:174, 118:177, 119:178, 120:178, 121:178, 122:178, 123:182, 124:182, 125:182, 126:186, 127:187, 128:188, 129:188, 130:189, 131:190, 132:191, 133:191, 134:192, 135:192, 136:193, 137:195, 138:195, 139:202, 140:203, 141:204, 142:204, 143:206, 144:206, 145:212, 146:213, 147:214, 148:217, 149:217, 150:218, 151:218, 152:225, 153:225, 154:225, 155:226, 156:229, 157:229, 158:229, 159:231, 160:236, 161:238, 162:238, 163:240, 164:240, 165:241, 166:241, 167:243, 168:245, 169:247, 170:248, 171:248, 172:248, 173:249, 174:250, 175:251, 176:252, 177:253, 178:255, 179:259, 180:262, 181:263, 182:265, 183:265, 184:266, 185:268, 186:270, 187:270, 188:275, 189:277, 190:277, 191:278, 192:279, 193:286, 194:291, 195:294, 196:297, 197:297, 198:299, 199:300

第三次：

a

0:1, 1:6, 2:6, 3:8, 4:11, 5:12, 6:13, 7:15, 8:18, 9:19, 10:26, 11:35, 12:40, 13:40, 14:41, 15:44, 16:50, 17:61, 18:61, 19:66, 20:74, 21:77, 22:81, 23:87, 24:91, 25:92, 26:101, 27:104, 28:114, 29:114, 30:115, 31:127, 32:127, 33:144, 34:145, 35:147, 36:148, 37:152, 38:160, 39:160, 40:162, 41:162, 42:168, 43:171, 44:172, 45:175, 46:176, 47:177, 48:178, 49:180, 50:183, 51:186, 52:188, 53:192, 54:196, 55:197, 56:198, 57:201, 58:202, 59:205, 60:206, 61:207, 62:208, 63:211, 64:218, 65:226, 66:231, 67:234, 68:236, 69:240, 70:243, 71:244, 72:246, 73:249, 74:250, 75:255, 76:255, 77:257, 78:258, 79:258, 80:258, 81:261, 82:264, 83:265, 84:265, 85:265, 86:268, 87:271, 88:271, 89:283, 90:283, 91:285, 92:287, 93:287, 94:289, 95:293, 96:293, 97:295, 98:300, 99:300

b

0:5, 1:8, 2:10, 3:12, 4:14, 5:15, 6:17, 7:21, 8:28, 9:28, 10:34, 11:41, 12:43, 13:43, 14:44, 15:49, 16:49, 17:54, 18:58, 19:62, 20:64, 21:65, 22:68, 23:70, 24:73, 25:76, 26:77, 27:77, 28:80, 29:91, 30:94, 31:96, 32:98, 33:98, 34:102, 35:104, 36:113, 37:125, 38:126, 39:126, 40:127, 41:132, 42:136, 43:136, 44:137, 45:137, 46:140, 47:153, 48:157, 49:158, 50:165, 51:169, 52:169, 53:172, 54:172, 55:173, 56:176, 57:177, 58:180, 59:181, 60:181, 61:181, 62:184, 63:186, 64:191, 65:191, 66:199, 67:202, 68:207, 69:208, 70:209, 71:211, 72:212, 73:212, 74:213, 75:216, 76:217, 77:220, 78:224, 79:224, 80:227, 81:233, 82:243, 83:250, 84:260, 85:261, 86:262, 87:264, 88:266, 89:274, 90:277, 91:280, 92:280, 93:282, 94:284, 95:289, 96:292, 97:293, 98:294, 99:296

c

0:1, 1:5, 2:6, 3:6, 4:8, 5:8, 6:10, 7:11, 8:12, 9:12, 10:13, 11:14, 12:15, 13:15, 14:17, 15:18, 16:19, 17:21, 18:26, 19:28, 20:28, 21:34, 22:35, 23:40, 24:40, 25:41, 26:41, 27:43, 28:43, 29:44, 30:44, 31:49, 32:49, 33:50, 34:54, 35:58, 36:61, 37:61, 38:62, 39:64, 40:65, 41:66, 42:68, 43:70, 44:73, 45:74, 46:76, 47:77, 48:77, 49:77, 50:80, 51:81, 52:87, 53:91, 54:91, 55:92, 56:94, 57:96, 58:98, 59:98, 60:101, 61:102, 62:104, 63:104, 64:113, 65:114, 66:114, 67:115, 68:125, 69:126, 70:126, 71:127, 72:127, 73:127, 74:132, 75:136, 76:136, 77:137, 78:137, 79:140, 80:144, 81:145, 82:147, 83:148, 84:152, 85:153, 86:157, 87:158, 88:160, 89:160, 90:162, 91:162, 92:165, 93:168, 94:169, 95:169, 96:171, 97:172, 98:172, 99:172, 100:173, 101:175, 102:176, 103:176, 104:177, 105:177, 106:178, 107:180, 108:180, 109:181, 110:181, 111:181, 112:183, 113:184, 114:186, 115:186, 116:188, 117:191, 118:191, 119:192, 120:196, 121:197, 122:198, 123:199, 124:201, 125:202, 126:202, 127:205, 128:206, 129:207, 130:207, 131:208, 132:208, 133:209, 134:211, 135:211, 136:212, 137:212, 138:213, 139:216, 140:217, 141:218, 142:220, 143:224, 144:224, 145:226, 146:227, 147:231, 148:233, 149:234, 150:236, 151:240, 152:243, 153:243, 154:244, 155:246, 156:249, 157:250, 158:250, 159:255, 160:255, 161:257, 162:258, 163:258, 164:258, 165:260, 166:261, 167:261, 168:262, 169:264, 170:264, 171:265, 172:265, 173:265, 174:266, 175:268, 176:271, 177:271, 178:274, 179:277, 180:280, 181:280, 182:282, 183:283, 184:283, 185:284, 186:285, 187:287, 188:287, 189:289, 190:289, 191:292, 192:293, 193:293, 194:293, 195:294, 196:295, 197:296, 198:300, 199:300

# 三、基本排序算法

分别对不同的整数序列进行选择、插入和冒泡排序算计进行从小到大进行排序，并计算排序所用的比较次数和移动次数。

要求：

1. 原始序列a按如下三种方式生成：
   1. 随机产生1~100之间的100个整数；
   2. 随机产生100个单调增的整数；
   3. 随机产生100个单调降的整数。
2. 显示a进行选择排序后的序列；
3. 显示排序所用的比较次数和数据移动次数（交换一次按三次计算）。

## 选择排序算法

程序代码：

void selection(int a[], int N, int \*com, int \*move){

    int i,j,k;

    int min;

    \*com = 0; \*move = 0;

    \_for(i,0,N-1){

        min = i;

        \_for(j,i+1,N){

            if(a[j] < a[min]){

                min = j;

            }

            (\*com)++;

        }

        if(i != min){

            swap(&a[i],&a[min]);

            (\*move) += 3;

        }

    }

}

## 插入排序算法

程序代码：

void insertion(int a[], int N, int \*com, int \*move){

    int i,j,k;

    int temp;

    \*com = 0; \*move = 0;

    \_for(i,1,N){

        temp = a[i];

        \_for(j,0,i){

            (\*com)++;

            if(temp < a[j]){

                \_for\_meq(k,i-1,j){

                    a[k+1] = a[k];

                    (\*move)++;

                }

                break;

            }

        }

        a[j] = temp;

    }

}

## 冒泡排序算法

程序代码：

void bubble(int a[], int N, int \*com, int \*move){

    int i,j;

    int temp;

    \*com = 0; \*move = 0;

    \_for\_meq(i,N-2,0){

        \_for(j,i,N-1){

            (\*com)++;

            if(a[j] > a[j+1]){

                swap(&a[j],&a[j+1]);

                (\*move) += 3;

            }

        }

    }

}

比较不同算法的运行结果：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 初始数据形式 | 算法统计参数 | 选择排序 | 插入排序 | 冒泡排序 |
| 100个1~100之间的随机数 | 比较次数 | 4950 | 2642 | 4950 |
| 移动次数 | 288 | 2403 | 7209 |
| 100个单调增的整数 | 比较次数 | 4950 | 4950 | 4950 |
| 移动次数 | 0 | 0 | 0 |
| 100个单调降的整数 | 比较次数 | 4950 | 110 | 4950 |
| 移动次数 | 159 | 4939 | 14817 |