数据结构与算法作业 1

算法设计（要求：算法用伪代码或 C/C++/Java 描述，并分析最坏情况下的时间复杂度）

1. 对一个整型数组 A[n]设计一个排序算法。
2. 找出整型数组 A[n]中元素的最大值和次最大值。
3. A 是一个有 n 个不同元素的实数数组，写出确定给定实数 K 是否在 A 中的算法，分析时间复杂性（要求：问题规模、 基本语句，O 记号），并在机器验证你的复杂度。
4. 设 A 是一个有 n 个不同元素的实数数组，写出求其最大和最小元素的算法，分析其时间复杂性。并在机器上验证你的复杂度。
5. 写出汉诺塔问题的求解算法，分析其时间复杂性。并在机器上验证你的复杂

答案：

1.

源代码：

```
#include <iostream>
using namespace std;
void xsort(int *psort, int n)
{
 int tep;
 int flag = 0;
 for(int i = 0; i < n-1; i++)
 {
    tep = psort[i];
    for(int j = i+1; j < n; j++)
    {
       if(psort[j] <  tep)
       {
          tep = psort[j];
          flag = j;
       }
    }
    if(flag)
       {
          psort[flag] = psort[i];
          psort[i] = tep;
       }
 }
}
```

```
int main()
{
 int a[10] = {1, 2, 8, 4, 2, 5, 8, 4, 2, 10};
 xsort(a, 10);
 for(int i = 0; i < 10; i++)
 cout << a[i] << " ";
 return 0;
}
```

最坏情况复杂度：O(n^2)--（两个循环）

2.

```
#include <iostream>
using namespace std;
void xfind(int *pa, int n)
{
 int tep = pa[0];
 int flag;
 for(int i = 0; i < n; i++)
 {
    if(pa[i] > tep)
    {
       tep = pa[i];
       flag = i;
    }
 }
 cout << tep << " ";
 tep = pa[0];
 for(int j = 0; j < n; j++)
 {
    if((pa[j] > tep) && (j != flag))
    {
       tep = pa[j];
    }
 }
 cout << tep << " ";

}

int main()
{
 int a[10] = {1,4,6,2,8,4,5,3,6,0};
 xfind(a, 10);
 return 0;
```

```
}
```

最坏时间复杂度 O(N)--（一个 for 循环）

3.

```
#include <iostream>
using namespace std;
int main()
{
    int i, j, flag = 0;
    int a[10] = {1,4,5,2,7,9,5,8,9,9};
    cin >> i;
    for(j = 0; j < 10; j++)
    {
        if(i == a[j])
        {
            flag = 1;
            cout << i <<"存在";
            break;
        }
    }
    if(!flag)
    {
        cout << i << "不存在";
    }
}
```

最坏时间复杂度 O(N)--（一个 for 循环）

4.

```
#include <iostream>
using namespace std;
int main()
{
    int a[10] = {1,4,5,7,3,4,6,9,0,3};
    int high = a[0], low = a[0];
    for(int i = 0; i < 10; i++)
    {
        if(high < a[i])
        {
            high  = a[i];
        }
        if(low > a[i])
        {
```

```
            low = a[i];
        }
    }
    cout << "最大值：" << high <<"\n 最小值" << low << "\n";
    return 0;
}
```

最坏时间复杂度：O(N)---（一个 for 循环）

5.

```
#include <iostream>
using namespace std;
void hanoi(int n, char a, char b, char c)
{
    if(n == 1)
    {
        cout << n << " " << a << " " << c << endl;
    }
    else
    {
        hanoi(n-1, a, c, b);
        cout  << n << " " << a << " " << c <<endl;
        hanoi(n-1, b, a, c);
    }
}

int main()
{
    int n;
    cin >> n;
    hanoi(n, 'A', 'B', 'C');
    return 0;
}
```

最坏时间复杂度（O(2^N)）--（迭代）