# 习题一

## 1.2

满足算法特点，理由如下：

* 有穷性：给出的算法有有限的步骤，且两个步骤都可以在有穷的时间内完成
* 可形性：列举全部子集，求和，比较等操作都是可执行的
* 确定性：两个步骤的描述没有二义性
* 输入：整数集S，整数m
* 输出：和为m的S的子集

## 1.5

|  |  |  |
| --- | --- | --- |
| **FindMax(A)** | **Cost** | **Times** |
| max←A[1] |  | 1 |
| For j←2 to n do |  | n |
| If A[j] > max than |  | n-1 |
| max ← A[j] |  | 0 to n-1 |
| return max |  | 1 |

所以：总的时间复杂度：

* 最坏情况下：
* 最好情况下：

都是输入规模n的线性函数

## 1.6

算法正确性证明：

1. 当j=1时，max=j[1]，显然正确
2. 当j=k时，若算法正确，则max为A[1…k]中的最大值，若A[k+1]>max，则max=A[k+1]，若A[k+1]≤max，则max为A[1…k]中的最大值。综上，max为A[1…k+1]中的最大值
3. 算法终止时j=n+1，由前面两步可以知道max为A[1…n]中的最大值

## 1.7

FindMaxIdx(A) //输出最后出现的最大元素的下标

max←A[1]

maxIdx←1

For j←2 to n do

If A[j] ≥ max than

max ← A[j]

maxIdx←j

return maxIdx

## 1.8

算法正确性证明

1. 循环开始前：i=1，pow=1，可以看成a的0次方，正确
2. 循环过程中，若i=k的时候算法正确，此时
3. 循环结束的时候，i=n+1，

因此算法正确

## 1.9

1 def Find(A, x):

2 for a in A:

3 if a == x:

4 return True

5 return False

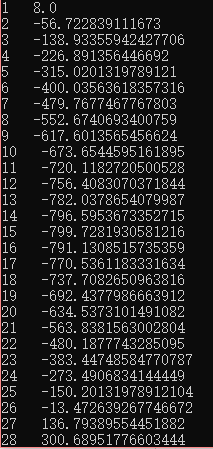
复杂度：算法最多需要遍历一次A，假设语句2，3的cost是c，语句4(5)的cost是v，则总的复杂度(花销)为cn+v

## 1.10

，可得n>14后，所以当n>14(n为整数)。

## 1.11

当1<n<27的时候：



# 编程作业

## 1001

 倒序输出一个四位整数

代码：

void func\_1001()

{

int a, N = 4, res = 0;

std::cin >> a;

for (int i = 1; i != N + 1; ++i) {

int tmp = (a % (int)pow(10, i)) / (int)pow(10, i - 1);

res += tmp \* (int)pow(10, N - i);

a -= tmp;

}

std::cout << res << std::endl;

}

## 1002

统计字符数

代码：

void func\_1002()

{

char tmp;

std::map<char, int> character\_counter;

while (std::cin >> tmp) {

if (tmp == '?') { break; }

++character\_counter[tmp];

}

for (char character : "123456789abcdefghijklmnopqrstuvwxyz") {

if (character) {

std::cout << character << ' ' << character\_counter[character] << '\n';

}

}

}