Q1 解：设若球不破碎测试楼层的序列为 $\{x_1, x_2, \cdots, x_n\}$，$x_0 = 0$

依题，当第1个球在 x_i 层破碎时，第2个球从 $x_{i-1}+1$ 层开始逐层测试，直至试破碎

故平均测试次数为 $\frac{1}{128}\sum_{j=1}^{n}\sum_{i=1}^{x_j-x_{j-1}-1}(j+i) = \frac{1}{128}\sum_{j=1}^{n}\frac{(x_j-x_{j-1}-1)(x_j-x_{j-1}+2j)}{2}$

而对于伪二分法 $\{x_1, \cdots, x_n\} = \{64, 96, 112, 120, 124, 126, 127, 128\}$　$n=8$

故 $E(f(x)) = \frac{1}{128}\sum_{j=1}^{n}\sum_{i=1}^{x_j-x_{j-1}-1}(j+i) = 22.546$

对于改良法 $\{x_1, \cdots, x_n\} = \{16, 31, 45, 58, 70, 81, 91, 100, 108, 115, 121, 126, 128\}$，$n=13$

故 $E(g(x)) = \frac{1}{128}\sum_{j=1}^{n}\sum_{i=1}^{x_j-x_{j-1}-1}(j+i) = 7.5625$

Q2：解：1. 正确，证明如下：

因为 $f(n) = \Theta(g(n))$ 所以存在常数 $c_1, c_2 > 0$ 与正整数 N_0，使得 $n \geq N_0$ 时有 $c_1 g(n) \leq f(n) \leq c_2 g(n)$。

故 $n \geq N_0$ 时，$\lg c_1 + \lg g(n) \leq \lg f(n) \leq \lg c_2 + \lg g(n)$

若 $c_2 \geq 1$ 取 $c_4 \geq \frac{\lg c_2}{\lg g(N_0)} + 1 \geq 1$，则 $(c_4-1)\lg g(n) \geq (c_4-1)\lg g(N_0) \geq \lg c_2$

所以有 $c_4 \lg g(n) \geq \lg c_2 + \lg g(n)$

若 $0 < c_2 < 1$，取 $c_4 > 1$，则 $c_4 \lg g(n) > \lg g(n) > \lg c_2 + \lg g(n)$

若 $c_1 \geq 1$，取 $0 < c_3 < 1$，则 $(c_3-1)\lg g(n) < 0 \leq \lg c_1$，故 $c_3 \lg g(n) \leq \lg c_1 + \lg g(n)$

若 $0 < c_1 < 1$，取 $c_3 \leq \frac{\lg c_1}{\lg g(N_0)} + 1 \leq \frac{\lg c_1}{\lg g(n)} + 1$ 故 $c_3 \lg g(n) \leq \lg c_1 + \lg g(n)$

综上，存在 c_3, c_4 使得当 $n \geq N_0$ 时　$c_3 \lg g(n) \leq \lg f(n) \leq c_4 \lg g(n)$

故 $\lg f(n) = \Theta(\lg g(n))$

2. 正确　证明如下：

易知 $\lim_{n\to\infty} \frac{b_i n^i}{b_k n^k} = 0$　$i = 0, 1, 2, \cdots, k-1$

故 $f(n) = \Theta(n^k)$

Q3. 解：依题，该计算机每个小时可执行的操作次数为　$2^{10} = 1024 \approx 10^3$　$2^{30} \approx 10^9$

$10^{10} \cdot 3600 = 3.6 \times 10^{13}$ 次

1. 令 $n^4 \leq 3.6\times10^{13} < (n+1)^4$ 得 $n \leq 2449.49 \leq n+1$，故 $n = 2449$

2. 令 $100n^2 \leq 3.6\times10^{13} \leq 100(n+1)^2$ 得 $n \leq 6\times10^5 \leq n+1$ 故 $n = 6\times10^5$

3. 令 $n\log n \leq 3.6\times10^{13} \leq (n+1)\log(n+1)$ 得 $n = 906316482883$

4. 令 $2^n \leq 3.6\times10^{13} \leq 2^{n+1}$ 得 $n \leq 45.03306 < n+1$，故 $n = 45$

5. 令 $2^{2^n} \leq 3.6\times10^{13} \leq 2^{2^{n+1}}$ 得 $n \leq 5.49291 \leq n+1$，故 $n = 5$

Q4. 依题

$f_1(n) = n^\pi = \Theta(n^\pi)$　$f_2(n) = \pi^n = \Theta(\pi^n)$　$f_3(n) = \binom{n}{5} = \frac{n!}{5!(n-5)!} = \Theta(n^5)$　$f_4(n) = \sqrt{2^{\sqrt{n}}} = \sqrt{2}^{\sqrt{n}} = \Theta(\sqrt{2}^{n^{\frac{1}{2}}})$

$f_5(n) = \binom{n}{n-4} = \binom{n}{4} = \frac{n!}{4!(n-4)!} = \Theta(n^4)$　$f_6(n) = 2^{\log^4 n}$　$f_7(n) = n^{5(\log n)^2}$

$$f_8(n) = n^4 \binom{n}{4} = \Theta(n^8)$$

根据对数函数<幂函数<指数函数(从增长速度看)

按照增长顺序排列为

$g_1(n) = f_1(n)$, $g_2(n) = f_5(n)$. $g_3(n) = f_3(n)$. $g_4(n) = f_8(n)$

$g_5(n) = f_6(n)$. $g_6(n) = f_7(n)$ $g_7(n) = f_4(n)$. $g_8(n) = f_2(n)$

Q5. 解: 1. 依题可得

i	1	2	3	4	5
a_i	1	3	3	5	2

故根据规则可知 $c_2 > c_1$, $c_4 > c_3$. $c_4 > c_5$

取 $c_1 = 1$. $c_2 = 2$, $c_3 = 2$, $c_4 = 3$. $c_5 = 2$ 即符合要求

2. 该算法的思路为

① 找到所有满足 $a_i \le a_{i+1}$. $a_i \le a_{i-1}$ 的 i (对于 $i=1$ 或 $i=n$. 只需满足一个条件), 记录在 min_id[ ] 中. 符合条件的 i 有 N 个

② 对①中每一个 i 记录 grade[i] = 1, 即 $c_i = 1$

③ 取 $i=1$ 令 num=1, j = min_id[i] 若有 a[j] > a[j-1], num++. 令 grade[j] = num, j++

若 a[j] = a[j-1], 令 grade[j] = num, j++

若 a[j] < a[j-1], break, i++ 直至 i=N+1

④ 取 i=N, 令 num=1, j = min_id[i] 若有 a[j] > a[j+1], num++. 令 grade[j] = num, j--

若 a[j] = a[j+1], 令 grade[j] = num, j--

若 a[j] < a[j+1], break, i-- 直至 i = 0

⑤ 此时 grade[i] 即为 c_i

该算法的C代码如下

```c
#include<stdio.h>

#define N 12
#define INFTY 100   // make sure that INFTY is bigger than a[i] for every i.

void FIND_MIN(int *list, int *min_id){  // find the id of local minimum in the list
    int i, j = 1;
    for(i = 1; i < N - 1; i++){
        if(list[i] <= list[i + 1] && list[i] <= list[i - 1]){
            min_id[j] = i;
            j++;
            min_id[0]++;
        }
    }
}

void PRINT_THE_LIST(int *list){         // as it's named, print the list
    for(int i = 0; i < N; i++){
        printf("%2d, ", list[i]);
    }
    printf("\n");
}

int main(){
    int list[N] = {INFTY, 1, 3, 4, 5, 3, 2, 6, 6, 6, 7, INFTY}, i, j, num = 1, INDEX[N] = {0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11};
    int min_id[N] = {0};    // min_id[0] means the number of the local minimum in the list, and min_id[1] - min_id[N-1] shows the details.
    int grade[N] = {0};     // grade[i] is c[i], actually
    FIND_MIN(list, min_id);
```

```
    for(i = 1; i <= min_id[0]; i++){    // from left to right
        for(j = min_id[i]; list[j] >= list[j - 1] || j == min_id[i]; j++){
            if(list[j] > list[j - 1]){
                grade[j] = ++num;
            }
            else{
                grade[j] = num;
            }
        }
        num = 1;
    }

    for(i = min_id[0]; i > 0; i--){     // from right to left
        for(j = min_id[i]; list[j] >= list[j + 1] || j == min_id[i]; j--){
            if(list[j] > list[j + 1]){
                num++;
                if(grade[j] < num){
                    grade[j] = num;
                }
            }
            else{
                if(grade[j] < num){
                    grade[j] = num;
                }
            }
        }
        num = 1;
    }

    PRINT_THE_LIST(min_id);
    PRINT_THE_LIST(INDEX);
    PRINT_THE_LIST(list);
    PRINT_THE_LIST(grade);
    return 0;
}
```