A：

递推：用 ans[m]来表示走 m 级阶梯有多少种走法，那么 ans[m]有两部分组成，第一是最后一步是“跨一级”来的，数量为 ans[m-1]，第二是最后一步是“跨两级”来的，数量为 ans[m-2]。所以 ans[m] = ans[m-1] + ans[m-2];

B:

DP：用 dp[i][j]来表示从第 i 行第 j 列的结点开始向底层走能得到的数字和的最大值。
由于从(i,j)结点向下走只有两种选择: 经过结点(i+1,j)，或者经过结点(i+1,j+1)。
所以状态转移方程为 dp[i][j] = dp[i+1][j] + dp[i+1][j+1];

C:

DP: 教材 P64 页有不止一种方法来解决该经典问题。

D:

贪心：教材 P40 页的区间调度问题。注意下：1.时间并非正整数，而是非负整数。2.开始结束瞬间可重叠。这两点样例都有提示的。

E:

贪心：The decimal representation of the final number shouldn't start with a zero.，是说结果没有前置 0，也就是说输入 98，应该输出 91，而不是 1。
做法就是：如果数字最高位是 9 的话，从次高位开始贪心的把较大的数字反转成较小的数字，否则就从最高位开始贪心的把较大的数字反转成较小的数字。
最后注意下数据范围有 10^18，int 会爆，long long 存答案就成了。

AC 代码：

A：

```
#pragma warning(disable:4996)
#include <cstdio>
using namespace std;
int ans[41];
int main() {
    ans[1] = ans[2] = 1;
    for (int i = 3; i <= 40; i++)
        ans[i] = ans[i - 1] + ans[i - 2];
    int n, m;
    scanf("%d", &n);
    while (n--) {
        scanf("%d", &m);
        printf("%d\n", ans[m]);
    }
    return 0;
}
```

B:

```
#pragma warning(disable:4996)
#include <cstdio>
#include <cstring>
#include <algorithm>
using namespace std;
int a[105][105];
int dp[105][105];
//dp[i][j]=a[i][j]+max(dp[i+1][j],dp[i+1][j+1]);
int main() {

    int C;
    scanf("%d", &C);
    while (C--) {
        int N;
        scanf("%d", &N);
        for (int i = 1; i <= N; i++) {
            for (int j = 1; j <= i; j++)
                scanf("%d", &a[i][j]);
        }

        memset(dp, 0, sizeof dp);
        for (int i = N; i > 0; i--) {
            for (int j = 1; j <= i; j++) {
                dp[i][j] = a[i][j] + max(dp[i + 1][j], dp[i + 1][j + 1]);
            }
        }
        printf("%d\n", dp[1][1]);
    }

    return 0;
}
```

C:

```
#pragma warning(disable:4996)
#include <cstdio>
#include <algorithm>
using namespace std;

int a[1005];
int dp[1005];
int main() {
    int n;
    scanf("%d", &n);
```

```
    for (int i = 1; i <= n; i++)scanf("%d", a + i);
    int ans = 0;
    for (int i = 1; i <= n; i++) {
        dp[i] = 1;
        for (int j = 1; j < i; j++) {
            if (a[i] > a[j])
                dp[i] = max(dp[i], dp[j] + 1);
        }
        ans = max(ans, dp[i]);
    }
    printf("%d\n", ans);
    return 0;
}
```

D:

```
#pragma warning(disable:4996)
#include <cstdio>
#include <algorithm>
using namespace std;

struct interval {
    int x, y;
    interval(){}
    bool operator<(const interval& op)const {
        if (y == op.y)return x>op.x;
        return y < op.y;
    }
};

interval a[105];

int main() {
    int n;
    //freopen("in.txt", "r", stdin);
    while (scanf("%d", &n) && n) {
        for (int i = 0; i < n; i++)scanf("%d%d", &a[i].x, &a[i].y);
        sort(a, a + n);

        int ans = 0, last = -1;
        for (int i = 0; i < n; i++) {
            if (a[i].x >= last) {
                ans++;
                last = a[i].y;
            }
```

```
        }
        printf("%d\n", ans);
    }
    return 0;
}
```

E:

```
#pragma warning(disable:4996)
#include <cstdio>
#include <cstring>
using namespace std;

char s[100];
int main() {
    scanf("%s", s);
    int len = strlen(s), i = 0;
    if (s[i] == '9')i = 1;

    for (; i < len; i++) {
        int num = s[i] - '0';
        if (num > 4) {
            s[i] = '0' + (9 - num);
        }
    }
    long long ans = 0;
    for (i = 0; i < len; i++) {
        ans *= 10;
        ans += s[i] - '0';
    }
    printf("%I64d\n", ans);
    return 0;
}
```