

A. 细胞游戏

无算法，考查编程基本功

```
1:  #include <iostream>           // jk
2:  #include <string.h>
3:  #include <algorithm>
4:
5:  using namespace std;
6:
7:  char board[105][105];
8:  int m, n, q;
9:
10: int checkLogic(int i, int j) {
11:     int count = 0;
12:     int left = j - 1;
13:     int right = j + 1;
14:     int top = i - 1;
15:     int bottom = i + 1;
16:     for(int x = top; x <= bottom; x++){
17:         for(int y = left; y <= right; y++){
18:             count = board[x][y] == 1 || board
[x][y] == -1 ? count + 1 : count;
19:         }
20:     }
21:     return board[i][j] == 1 ? (count == 3 ||
count == 4 ? 1 : -1) : (count == 3 ? -2 : 0);
22: }
23:
24: void update() {
25:     for(int i = 1; i <= m; i++){
26:         for(int j = 1; j <= n; j++){
27:             board[i][j] = checkLogic(i, j);
28:         }
29:     }
```

```

30:     for(int i = 1; i <= m; i++){
31:         for(int j = 1; j <= n; j++){
32:             board[i][j] = board[i][j] == 1 ||
board[i][j] == -2 ? 1 : 0;
33:         }
34:     }
35: }
36:
37: int main() {
38:     cin >> m >> n >> q;
39:     memset(board, 0, sizeof(board));
40:     for (int i = 1; i <= m; ++i) {
41:         for (int j = 1; j <= n; ++j) {
42:             cin >> board[i][j];
43:             board[i][j] -= '0';
44:         }
45:     }
46:
47:     while (q--) update();
48:
49:     for (int i = 1; i <= m; ++i) {
50:         for (int j = 1; j <= n; ++j) {
51:             cout << (int)board[i][j] << " \n"
[j==n];
52:         }
53:     }
54:     return 0;
55: }

```

B. 嵌套深度

常规写法:

```
1:  #include<bits/stdc++.h>           // Enal
2:
3:  using namespace std;
4:
5:  int main()
6:  {
7:      int n;
8:      scanf("%d",&n);
9:      for(int i=1;i<=n;++i){
10:         char c[200];
11:         scanf("%s",c);
12:         int cnt=0,maxnum=0;
13:         for(int j=0;j<strlen(c);++j){
14:             if(c[j]=='('){
15:                 ++cnt;
16:             }else{
17:                 --cnt;
18:             }
19:             if(cnt<0){
20:                 puts("-1");
21:                 break;
22:             }else{
23:                 maxnum=max(cnt,maxnum);
24:             }
25:         }
26:         if(cnt==0){
27:             cout<<maxnum-1<<endl;
28:         }else if(cnt>0){
29:             puts("-1");
30:         }
31:     }
```

```
32:     return 0;
33: }
```

用栈写提示：

- (1) #号作为开始和结束的标志。
- (2) 出现的凡是“左括号”，则进栈。
- (3) 出现的是“右括号”

首先检查栈是否空？

- 1. 若栈空，则表明该“右括号”多余
- 2. 否则和栈顶元素比较？
 - 2.1 若相匹配，则栈顶“左括号出栈”
 - 2.2 否则表明不匹配

(4) 表达式检验结束时

- 1. 若栈空，则表明表达式中匹配正确
- 2. 否则表明“左括号”有余

C. 装水容器

$n \leq 100$, 暴力 $O(n^2)$:

```
1:  # include <iostream>    // kcxz
2:
3:  int a[105];
4:
5:  int main() {
6:      int T;
7:      std::cin >> T;
8:
9:      while (T--) {
10:         int n;
11:         std::cin >> n;
12:         for (int i=0; i<n; i++) {
13:             std::cin >> a[i];
14:         }
15:
16:         int t = 0;
17:         int max = 0;
18:         for (int i=0; i<n; i++) {
19:             for (int j=i+1; j<n; j++) {
20:                 t = std::min(a[j], a[i]) * (j
21: -i);
22:                 if (t > max) {
23:                     max = t;
24:                 }
25:             }
26:         }
27:         std::cout << max << std::endl;
28:     }
29:
30:     return 0; }
```

O(n) 算法:

```
1:  # include <iostream>    // kcxz
2:
3:  int a[105];
4:
5:  int main() {
6:      int T;
7:      std::cin >> T;
8:
9:      while (T--) {
10:         int n;
11:         std::cin >> n;
12:         for (int i=0; i<n; i++) {
13:             std::cin >> a[i];
14:         }
15:
16:         int t = 0;
17:         int max = 0;
18:         int left = 0;
19:         int right = n-1;
20:         for (int i=0; i<n; i++) {
21:             if (left > right) {
22:                 break;
23:             }
24:
25:             t = std::min(a[left], a[right]) *
(right - left);
26:             if (t > max) {
27:                 max = t;
28:             }
29:
30:             if (a[left] < a[right]) {
31:                 left++;
32:             } else {
33:                 right--;
```

```
34:         }
35:     }
36:
37:     std::cout << max << std::endl;
38: }
39:
40: return 0;
41: }
```

D. 大佬数

质因数，即既是因数又是质数。

算术基本定理：

任何一个大于 1 的自然数N, 如果N 不为质数，那么N 可以唯一分解成有限个质数的乘积。

```
1:  # include <iostream>
2:
3:  int main() {
4:      int n;
5:      std::cin >> n;
6:
7:      int sum = 0;
8:      for (int i=2; i<=n; i++) {
9:          if (0 == n%i) {
10:             sum += i;
11:
12:             while (0 == n%i) {
13:                 n /= i;
14:             }
15:         }
16:     }
17:
18:     if (sum > 610) {
19:         std::cout << "a sdl wsl";
20:     } else {
21:         std::cout << "tcl";
22:     }
23:
24:     return 0;
25: }
```


暴力法:

```
1:  # include <iostream>    // kcxz
2:
3:  bool isPrime(int n) {
4:      if (n <= 1)
5:          return false;
6:
7:      if (n <= 3)
8:          return true;
9:
10:     if (n % 2 == 0 || n % 3 == 0)
11:         return false;
12:
13:     for (int i = 5; i * i <= n; i = i + 6)
14:         if (n % i == 0 || n % (i + 2) == 0)
15:             return false;
16:
17:     return true;
18: }
19:
20: int main() {
21:     int n;
22:     std::cin >> n;
23:
24:     int cnt = 0;
25:     for(int i=2; i<=n; i++) {
26:         if (n % i == 0) {
27:             if (isPrime(i)) {
28:                 cnt += i;
29:             }
30:         }
31:     }
32:
33:     if (cnt > 610) {
```

```
34:         std::cout << "a sdl wsl";
35:     } else {
36:         std::cout << "tcl";
37:     }
38:
39:     return 0;
40: }
```

E. 能不能让我捧个杯啊

无算法，考查编程基本功

```
1:  # include <iostream>    // kcxz ?
2:  # include <cstdio>
3:
4:  bool vis[705];
5:  int pos[705];
6:
7:  int main(){
8:      int n;
9:      int k;
10:     scanf("%d %d", &n, &k);
11:
12:     int x;
13:     for (int i=1; i<=k; i++) {
14:         scanf("%d", &x);
15:         vis[x] = true;
16:     }
17:
18:     int j = 0;
19:     for (int i=1; i<=n+k; i++) {
20:         if(!vis[i]) {
21:             pos[++j] = i;
22:         }
23:     }
24:
25:     n /= 10;
26:     printf("%d %d\n%d %d\n%d %d\n", pos[1], pos[n], pos[n+1], pos[n*3], pos[n*3+1], pos[n*6]);
27:
28:     return 0;
29: }
```

F. 多组输出斐波拉契

考查斐波拉契数列和模运算。

根据求模公式：

$$(a + b) \% c = (a \% c + b \% c) \% c$$

从而有：

$$F_n \% m = F_{n-1} \% m + F_{n-2} \% m = (F_{n-1} + F_{n-2}) \% m$$
$$(n \geq 3 \ \&\& \ n \in \mathbb{N})$$

只需要将每一项斐波拉契求模即可。

只有一秒的时间，需要将数据记忆下来，否则还是会超时。

另外，数据过大，请不要用 java 写，也不要 C++ 中的 cin 和 cout

复杂度： $O(n)$

```
1: # include <iostream>    // kcxz
2: # include <cstdio>
3:
4: const int mod = 998244353;
5: const int maxn = 1e6+5;
6: int a[maxn];
7:
8: void init() {
9:     a[0] = 0;
10:    a[1] = 1;
11:    a[2] = 1;
12:    for (int i=3; i<maxn; i++) {
13:        a[i] = (a[i-1] + a[i-2]) % mod;
14:    }
15: }
16:
17: int main() {
18:     init();
19:     int T;
```

```

20:     scanf("%d", &T);
21:
22:     while (T--) {
23:         int t;
24:         scanf("%d", &t);
25:
26:         printf("%d\n", a[t]);
27:     }
28:
29:     return 0;
30: }

```

矩阵快速幂： 请自行查阅相应知识
复杂度： $O(n \log n)$

```

1: // O(nLogn)
2: #include <iostream>
3:
4: template <typename T>
5: void read(T &val) {
6:     T x = 0;
7:     int bz = 1;
8:     char c;
9:
10:    for (c = getchar(); (c < '0' || c > '9') && c != '-';
11:         c = getchar());
12:
13:    if (c == '-') {
14:        bz = -1;
15:        c = getchar();
16:    }
17:
18:    for (; c >= '0' && c <= '9'; c = getchar())
19:        x = x * 10 + c - 48;
20:
21:    val = x * bz;
22: }
23:
24: template <typename T>
25: void put(T x){
26:     static char s[20];
27:     int bas;
28:
29:     if(x < 0) {
30:         putchar('-');

```

```

31:         x = -x;
32:     }
33:
34:     if(x == 0) {
35:         putchar('0');
36:         return;
37:     }
38:
39:     bas = 0;
40:     for(;x;x/=10)
41:         s[bas++] = x%10+'0';
42:
43:     for(;bas--;)
44:         putchar(s[bas]);
45: }
46:
47: const int mod = 998244353;
48: struct Node {
49:     int a[2][2] = {{1,1},{1,0}};
50:     Node& operator* (Node& b ) {
51:         Node x;
52:         x.a[0][0] = 111 * (111 * this->a[0][0] * b.a[0][0] ) % mod + (111 * this->a[0][1] * b.a[1][0] ) % mod;
53:         x.a[0][1] = 111 * (111 * this->a[0][0] * b.a[0][1] ) % mod + (111 * this->a[0][1] * b.a[1][1] ) % mod;
54:         x.a[1][0] = 111 * (111 * this->a[1][0] * b.a[0][0] ) % mod + (111 * this->a[1][1] * b.a[1][0] ) % mod;
55:         x.a[1][1] = 111 * (111 * this->a[1][0] * b.a[0][1] ) % mod + (111 * this->a[1][1] * b.a[1][1] ) % mod;
56:
57:         for (int i=0; i<2; i++) {
58:             for (int j=0; j<2; j++) {
59:                 a[i][j] = x.a[i][j];
60:             }
61:         }
62:
63:         return *this;
64:     }
65: };
66:
67: Node qpow(Node x, int p) {
68:     Node ans;
69:     ans.a[0][0] = 1;
70:     ans.a[0][1] = 0;
71:     ans.a[1][0] = 0;
72:     ans.a[1][1] = 1;
73:
74:     while (p) {
75:         if (p & 1) {
76:             ans = ans * x;
77:         }
78:         x = x * x;

```

```
79:         p >= 1 ;
80:     }
81:
82:     return ans;
83: }
84:
85: void solve(int n) {
86:     if (n == 1 || n == 2){
87:         put(1);
88:         puts("");
89:         return;
90:     }
91:
92:     Node a;
93:     a = qpow(a, n-1);
94:     put(a.a[0][0] % mod);
95:     puts("");
96: }
97:
98: int main() {
99:     int T;
100:    read(T);
101:
102:    while (T--) {
103:        int n;
104:        read(n);
105:
106:        solve(n);
107:    }
108:
109:    return 0;
110: }
```

G. 吃面包

考查 01 背包 (0-1 Knapsack) 方案数

```
1:  # include <iostream>    // NQ
2:  # include <cstdio>
3:  # include <algorithm>
4:
5:  using namespace std;
6:
7:  const int N = 1010, mod = 1e9 + 7;
8:  int n, m, nums[N];
9:
10: int main()
11: {
12:     scanf("%d %d", &n, &m);
13:     nums[0] = 1;
14:     while (n--) {
15:         int v;
16:         scanf("%d", &v);
17:         for (int i = m; i >= v; i--)
18:             nums[i] = (nums[i] + nums[i - v])
19: % mod;
20:     }
21:     cout << nums[m] << endl;
22:     return 0;
23: }
```


H. 能否购买

考查单调队列

```
1: # include <iostream>    // NQ
2: # include <cstdio>
3: # include <algorithm>
4:
5: using namespace std;
6:
7: const int N = 1e6 + 5;
8: int n, m, k, sum[N];
9: int q[N], hh, tt;
10:
11: int main()
12: {
13:     scanf("%d %d %d", &n, &m, &k);
14:     for (int i = 1; i <= n; i++) {
15:         scanf("%d", &sum[i]);
16:         sum[i] += sum[i - 1];
17:     }
18:     int max_res = -0x3f3f3f3f;
19:     for (int i = 1; i <= n; i++) {
20:         if (hh <= tt && i - q[hh] > k)
21:             hh++;
22:         max_res = max(max_res, sum[i] - sum[q[hh]]);
23:         while (hh <= tt && sum[i] <= sum[q[tt]])
24:             tt--;
25:         q[++tt] = i;
26:     }
27:     while (m--) {
28:         int x;
29:         scanf("%d", &x);
```

```
30:         if (max_res >= x)
31:             printf("YES\n");
32:         else
33:             printf("NO\n");
34:     }
35:     return 0;
36: }
```