



初赛模赛

选择题

快快编程
kkcoding.net

选择

1. 一幅 256×256 的图像，若灰度级数为 16，则存储它所需的比特数是()。

A. 256kbit

B. 512kbit

C. 1M

D. 2M

答案：A

解释： 256×256 表示像素个数，16级灰度用二进制表示需要 4 位，故存储图像所需的二进制位数为： $256 \times 256 \times 4 = 2^{18} \text{bit} = 256 \text{kbit} = 32 \text{kbyte}$

选择

2. 下面关于图的论述中哪个是不正确的 ()?
- A. 图由顶点和边的集合组成
 - B. 有向图中不允许存在没有边或边连接的顶点存在
 - C. 图中所有顶点的度之和为边的数量的两倍
 - D. 邻接表是图的一种链式存储结构

答案：B

解释：有向图只有一个节点时没有边连接

选择

3. 用两种颜色去围城一个圈的 6 个棋子，如果通过旋转得到则只算一种，

则一共有多少 () 种染色模式。

A.10

B.14

C.15

D.16

答案： B

解释： $n(0)=n(6)=1$ ；全黑或全白， $n(1)=n(5)=1$ ；有一个黑色棋子，只能有一种组合，旋转相同， $n(2)=n(4)=3$ ；分别是两个相邻，相隔一个，相隔两个， $n(3)=4$ ；分别是 BBB,BBWB,BBWWB,BWBWB $1*2+1*2+3*2+4=14$

选择

4. 为了保证公司网络的安全运行，预防计算机病毒的破坏，可以在计算机上采取以下哪种方法 ()。

- A. 磁盘扫描
- B. 安装浏览器加载项
- C. 开启防病毒软件
- D. 修改注册表

答案：C

解释：防病毒软件是一种计算机程序，可进行检测、防护，并采取行动来解除或删除恶意软件程序，如病毒和蠕虫

选择

5. 十进制小数为 0.96875，对应的二进制数为 ()。

A.0.11111

B.0.111101

C.0.111111

D.0.1111111

答案：A

解释：十进制小数转换成二进制小数采用“乘 2 取整，顺序排列”法。具体做法是：用 2 乘十进制小数，可以得到积，将积的整数部分取出，再用 2 乘余下的小数部分，又得到一个积，再将积的整数部分取出，如此进行，直到积中的小数部分为零，此时 0 或 1 为二进制的最后一位

选择

6. 由 3 个“1”和 5 个“0”组成的 8 位二进制补码，能表示的最小整数

()。

A.-126

B.-125

C.-32

D.-3

答案：B

解释：补码整数表示时，负数的符号位为 1，数值位按位取反，末位加 1，因此剩下的 2 个“1”在最低位时，表示的是最小整数，为 10000011，转换成真值为 -125

选择

7. 安迪-比尔定理指出: 硬件提高的性能, 很快被 () 消耗掉了。

A. 操作系统

B. CPU

C. 计算机病毒

D. 软件

答案: D

解释: 安迪-比尔定理是指硬件提高的性能, 很快被软件消耗掉了

选择

8. 某二叉树的先序遍历序列和后序遍历序列正好相反，则该二叉树具有的特征是 ()。

- A. 高度等于其结点数
- B. 任一结点无左孩子
- C. 任一结点无右孩子
- D. 空或只有一个结点

答案：A

解释：只要是链，不一定非要无左孩子或无右孩子。即高度和结点数相等

选择

9. 一个有序数列，序列中的每一个值都能够被 2 或者 3 或者 5 所整除，这个序列的初始值从 1 开始，但是 1 并不在这个数列中。求第 1500 个值是多少 ()。

A.2040

B.2042

C.2045

D.2050

答案：C

解释：2、3、5 的最小公倍数是 30。[1, 30] 内符合条件的数有 22 个。如果能看出 [31, 60] 内也有 22 个符合条件的数，那问题就容易解决了。也就是说，这些数具有周期性，且周期为 30。第 1500 个数是： $1500/22=68$
 $1500\%68=4$ 。也就是说：第 1500 个数相当于经过了 68 个周期，然后再取下一个周期内的第 4 个数。一个周期内的前 4 个数：2,3,4,5。故，结果为 $68*30=2040+5=2045$

选择

10. 云计算是通过大量在云端的计算资源进行计算，下列（ ）项不是云计算的特点。

- A. 超大规模
- B. 高可扩展性
- C. 无危险性
- D. 虚拟化

答案：C

解释：云计算中的数据对于数据所有者以外的其他用户云计算用户是保密的，但是对于提供云计算的商业机构而言确实毫无秘密可言。所以还是具有潜在的危险

选择

11. 假定利用数组 $a[n]$ 顺序存储一个栈，用 top 表示栈顶指针，用 $top == -1$ 表示栈空，并已知栈未满，当元素 x 进栈时所执行的操作为 ()。

- A. $a[--top] = x$
- B. $a[top--] = x$
- C. $a[++top] = x$
- D. $a[top++] = x$

答案：C

解释：先将指针 top 加 1，再将元素入栈

选择

12. 已知声明并初始化二维数组 “int a[3][2]={1,2},{3,4},{5,6};”，则 a[1][1] 的值为 ()。

A.1

B.2

C.4

D.5

答案：C

解释：数组初始化从0，0开始

快
kkcoding.net

选择

13. 十进制数 1000 对应的二进制数和十六进制数为 ()。

A.1111101010 和 3F8

B.1111101000 和 3E8

C.1111101100 和 3D8

D.1111101110 和 3C8

答案：B

解释：进制转换

选择

14. 设有一个递归算法如下。试问计算 $f(f(9))$ 时需要计算 () 次 f 函数。

```
int f(int n){  
    if(n<=3)  
        return 1;  
    else  
        return f(n-2)+f(n-6)+1;  
}
```

A.10 B.11 C.12 D.14

答案：C

解释：手算样例一遍

选择

15. 编译器的主要功能是 ()。
- A. 将一种高级语言翻译成另一种高级语言
 - B. 将源程序翻译成指令
 - C. 将低级语言翻译成高级语言
 - D. 将源程序重新组合

答案： B

解释： 编译器的功能是将源程序翻译成指令

阅读程序

快快编程
kkcoding.net

阅读程序

```
1  #include <bits/stdc++.h>
2  using namespace std;
3  int main() {
4      int n,k=1;
5      cin>>n;
6      while (n>k) {
7          n=n-k;
8          k++;
9      }
10     if(k%2==0) cout<<n<<"/"<<(k+1-n);
11     else cout<<k+1-n<<"/"<<n;
12     return 0;
13 }
```

阅读程序

```
1  #include <bits/stdc++.h>
2  using namespace std;
3  int main() {
4      int n,k=1;
5      cin>>n;
6      while (n>k) {
7          n=n-k;
8          k++;
9      }
10     if(k%2==0) cout<<n<<"/"<<(k+1-n);
11     else cout<<k+1-n<<"/"<<n;
12     return 0;
13 }
```

判断

1.程序输出的时候, $k+1-n$ 必定小于 n 。()

阅读程序

```
1  #include <bits/stdc++.h>
2  using namespace std;
3  int main() {
4      int n,k=1;
5      cin>>n;
6      while (n>k) {
7          n=n-k;
8          k++;
9      }
10     if(k%2==0) cout<<n<<"/"<<(k+1-n);
11     else cout<<k+1-n<<"/"<<n;
12     return 0;
13 }
```

判断	2.输出一定是一个真分数。()
----	------------------

阅读程序

```
1  #include <bits/stdc++.h>
2  using namespace std;
3  int main() {
4      int n,k=1;
5      cin>>n;
6      while (n>k) {
7          n=n-k;
8          k++;
9      }
10     if(k%2==0) cout<<n<<"/"<<(k+1-n);
11     else cout<<k+1-n<<"/"<<n;
12     return 0;
13 }
```

判断

3. k的初始值设置为0不影响程序结果。()

阅读程序

```
1  #include <bits/stdc++.h>
2  using namespace std;
3  int main() {
4      int n,k=1;
5      cin>>n;
6      while (n>k) {
7          n=n-k;
8          k++;
9      }
10     if(k%2==0) cout<<n<<"/"<<(k+1-n);
11     else cout<<k+1-n<<"/"<<n;
12     return 0;
13 }
```

判断

4. while判断变成 $n \geq k$, 程序结果会受到影响。()

阅读程序

```
1  #include <bits/stdc++.h>
2  using namespace std;
3  int main() {
4      int n,k=1;
5      cin>>n;
6      while (n>k) {
7          n=n-k;
8          k++;
9      }
10     if(k%2==0) cout<<n<<"/"<<(k+1-n);
11     else cout<<k+1-n<<"/"<<n;
12     return 0;
13 }
```

选择	5. 若输入7，程序输出结果为 ()。 A. 1/4 B. 2/4 C. 2/3 D. 1/3
----	---

阅读程序

```
1  #include <bits/stdc++.h>
2  using namespace std;
3  int main() {
4      int n,k=1;
5      cin>>n;
6      while (n>k) {
7          n=n-k;
8          k++;
9      }
10     if(k%2==0) cout<<n<<"/"<<(k+1-n);
11     else cout<<k+1-n<<"/"<<n;
12     return 0;
13 }
```

选择

6. 由程序可以看出，当n在什么范围时，分子分母之和为6（ ）
A. [1,2] B. [5,7] C. [11,15] D. [16,21]

阅读程序

快快编程
kkcoding.net

2分钟

阅读程序

```
1  #include <bits/stdc++.h>
2  using namespace std;
3  int a[1100],p;
4  int dp[1010];
5  int dfs(int pos,bool limit) {
6      if (pos==-1) return 1;
7      if (!limit && dp[pos]!=-1) return dp[pos];
8      int u = limit?a[pos]:9;
9      int ret = 0;
10     for (int i=0; i<=u; ++i) {
11         if (i==7) continue;
12         ret += dfs(pos-1,limit && i==a[pos]);
13     }
14     if (!limit) dp[pos] = ret;
15     return ret;
16 }
17 int work(int x) {
18     p = 0;
19     while (x) {
20         a[p++] = x%10;
21         x /= 10;
22     }
23     return dfs(p-1,true);
24 }
25 int main() {
26     memset(dp,-1,sizeof(dp));
27     int a = 0;
28     scanf("%d",&a);
29     printf("%d\n",work(a)-1);
30     return 0;
31 }
```

- 1.请提取所有变量
- 2.选择题中手算样例
3. 结合自己的经验，分析

2分钟

阅读程序

```
1 #include <bits/stdc++.h>
2 using namespace std;
3 int a[1100],p;
4 int dp[1010];
5 int dfs(int pos,bool limit) {
6     if (pos==-1) return 1;
7     if (!limit && dp[pos]!=-1) return dp[pos];
8     int u = limit?a[pos]:9;
9     int ret = 0;
10    for (int i=0; i<=u; ++i) {
11        if (i==7) continue;
12        ret += dfs(pos-1,limit && i==a[pos]);
13    }
14    if (!limit) dp[pos] = ret;
15    return ret;
16 }
17 int work(int x) {
18     p = 0;
19     while (x) {
20         a[p++] = x%10;
21         x /= 10;
22     }
23     return dfs(p-1,true);
24 }
25 int main() {
26     memset(dp,-1,sizeof(dp));
27     int a = 0;
28     scanf("%d",&a);
29     printf("%d\n",work(a)-1);
30     return 0;
31 }
```

选择

5. 若输入

10

输出结果是()。

A. 9 B. 10 C. 8 D. 7

手算样例

2分钟

阅读程序

```
1 #include <bits/stdc++.h>
2 using namespace std;
3 int a[1100],p;
4 int dp[1010];
5 int dfs(int pos,bool limit) {
6     if (pos==-1) return 1;
7     if (!limit && dp[pos]!=-1) return dp[pos];
8     int u = limit?a[pos]:9;
9     int ret = 0;
10    for (int i=0; i<=u; ++i) {
11        if (i==7) continue;
12        ret += dfs(pos-1,limit && i==a[pos]);
13    }
14    if (!limit) dp[pos] = ret;
15    return ret;
16 }
17 int work(int x) {
18     p = 0;
19     while (x) {
20         a[p++] = x%10;
21         x /= 10;
22     }
23     return dfs(p-1,true);
24 }
25 int main() {
26     memset(dp,-1,sizeof(dp));
27     int a = 0;
28     scanf("%d",&a);
29     printf("%d\n",work(a)-1);
30     return 0;
31 }
```

判断

1. 19-22程序在做数位分离并放到a数组中。()

2分钟

阅读程序

```
1 #include <bits/stdc++.h>
2 using namespace std;
3 int a[1100],p;
4 int dp[1010];
5 int dfs(int pos,bool limit) {
6     if (pos==-1) return 1;
7     if (!limit && dp[pos]!=-1) return dp[pos];
8     int u = limit?a[pos]:9;
9     int ret = 0;
10    for (int i=0; i<=u; ++i) {
11        if (i==7) continue;
12        ret += dfs(pos-1,limit && i==a[pos]);
13    }
14    if (!limit) dp[pos] = ret;
15    return ret;
16 }
17 int work(int x) {
18     p = 0;
19     while (x) {
20         a[p++] = x%10;
21         x /= 10;
22     }
23     return dfs(p-1,true);
24 }
25 int main() {
26     memset(dp,-1,sizeof(dp));
27     int a = 0;
28     scanf("%d",&a);
29     printf("%d\n",work(a)-1);
30     return 0;
31 }
```

判断

2. dp数组初始值为0。()

2分钟

阅读程序

```
1 #include <bits/stdc++.h>
2 using namespace std;
3 int a[1100],p;
4 int dp[1010];
5 int dfs(int pos,bool limit) {
6     if (pos==-1) return 1;
7     if (!limit && dp[pos]!=-1) return dp[pos];
8     int u = limit?a[pos]:9;
9     int ret = 0;
10    for (int i=0; i<=u; ++i) {
11        if (i==7) continue;
12        ret += dfs(pos-1,limit && i==a[pos]);
13    }
14    if (!limit) dp[pos] = ret;
15    return ret;
16 }
17 int work(int x) {
18     p = 0;
19     while (x) {
20         a[p++] = x%10;
21         x /= 10;
22     }
23     return dfs(p-1,true);
24 }
25 int main() {
26     memset(dp,-1,sizeof(dp));
27     int a = 0;
28     scanf("%d",&a);
29     printf("%d\n",work(a)-1);
30     return 0;
31 }
```

判断

3. dp数组作用是记忆化，提高了递归的效率。()

2分钟

阅读程序

```
1 #include <bits/stdc++.h>
2 using namespace std;
3 int a[1100],p;
4 int dp[1010];
5 int dfs(int pos,bool limit) {
6     if (pos==-1) return 1;
7     if (!limit && dp[pos]!=-1) return dp[pos];
8     int u = limit?a[pos]:9;
9     int ret = 0;
10    for (int i=0; i<=u; ++i) {
11        if (i==7) continue;
12        ret += dfs(pos-1,limit && i==a[pos]);
13    }
14    if (!limit) dp[pos] = ret;
15    return ret;
16 }
17 int work(int x) {
18     p = 0;
19     while (x) {
20         a[p++] = x%10;
21         x /= 10;
22     }
23     return dfs(p-1,true);
24 }
25 int main() {
26     memset(dp,-1,sizeof(dp));
27     int a = 0;
28     scanf("%d",&a);
29     printf("%d\n",work(a)-1);
30     return 0;
31 }
```

判断

4. 20行p++改为++p不会影响程序结果。()

2分钟

阅读程序

```
1 #include <bits/stdc++.h>
2 using namespace std;
3 int a[1100],p;
4 int dp[1010];
5 int dfs(int pos,bool limit) {
6     if (pos==-1) return 1;
7     if (!limit && dp[pos]!=-1) return dp[pos];
8     int u = limit?a[pos]:9;
9     int ret = 0;
10    for (int i=0; i<=u; ++i) {
11        if (i==7) continue;
12        ret += dfs(pos-1,limit && i==a[pos]);
13    }
14    if (!limit) dp[pos] = ret;
15    return ret;
16 }
17 int work(int x) {
18     p = 0;
19     while (x) {
20         a[p++] = x%10;
21         x /= 10;
22     }
23     return dfs(p-1,true);
24 }
25 int main() {
26     memset(dp,-1,sizeof(dp));
27     int a = 0;
28     scanf("%d",&a);
29     printf("%d\n",work(a)-1);
30     return 0;
31 }
```

选择

6. 若输入

12345

输出结果是 ()

A. 7670 B. 8303

C. 7499 D. 8534

阅读程序

快快编程
kkcoding.net

3分钟

阅读程序

```
1 #include <bits/stdc++.h>
2 using namespace std;
3 stack<int> n;
4 char ch;
5 int s,x,y;
6 int main()
7 {
8     while(ch!='@')
9     {
10         ch=getchar();
11         switch(ch)
12         {
13             case '+':x=n.top();n.pop();y=n.top();n.pop();n.push(x+y);break;
14             case '-':x=n.top();n.pop();y=n.top();n.pop();n.push(y-x);break;
15             case '*':x=n.top();n.pop();y=n.top();n.pop();n.push(x*y);break;
16             case '/':x=n.top();n.pop();y=n.top();n.pop();n.push(y/x);break;
17             case '.':n.push(s);s=0;break;
18             default :s=s*10+ch-'0';break;
19         }
20     }
21     printf("%d\n",n.top());
22     return 0;
23 }
```

- 1.请提取所有变量
- 2.选择题中手算样例
- 3.结合自己的经验，分析

3分钟

阅读程序

```
1 #include <bits/stdc++.h>
2 using namespace std;
3 stack<int> n;
4 char ch;
5 int s,x,y;
6 int main()
7 {
8     while(ch!='@')
9     {
10         ch=getchar();
11         switch(ch)
12         {
13             case '+':x=n.top();n.pop();y=n.top();n.pop();n.push(x+y);break;
14             case '-':x=n.top();n.pop();y=n.top();n.pop();n.push(y-x);break;
15             case '*':x=n.top();n.pop();y=n.top();n.pop();n.push(x*y);break;
16             case '/':x=n.top();n.pop();y=n.top();n.pop();n.push(y/x);break;
17             case '.':n.push(s);s=0;break;
18             default :s=s*10+ch-'0';break;
19         }
20     }
21     printf("%d\n",n.top());
22     return 0;
23 }
```

选择

3. 输入

3.5.2.-*7.+@

输出是 ()。

A. 15

B. 16

C. 17

D. 18

手算样例

3分钟

阅读程序

```
1 #include <bits/stdc++.h>
2 using namespace std;
3 stack<int> n;
4 char ch;
5 int s,x,y;
6 int main()
7 {
8     while(ch!='@')
9     {
10         ch=getchar();
11         switch(ch)
12         {
13             case '+':x=n.top();n.pop();y=n.top();n.pop();n.push(x+y);break;
14             case '-':x=n.top();n.pop();y=n.top();n.pop();n.push(y-x);break;
15             case '*':x=n.top();n.pop();y=n.top();n.pop();n.push(x*y);break;
16             case '/':x=n.top();n.pop();y=n.top();n.pop();n.push(y/x);break;
17             case '.':n.push(s);s=0;break;
18             default :s=s*10+ch-'0';break;
19         }
20     }
21     printf("%d\n",n.top());
22     return 0;
23 }
```

选择

4. 输入字符 ‘.’的作用是
()。

- A. 小数点
- B. 区分两个符号
- C. 区分两个数字
- D. 区分数字与符号

3分钟

阅读程序

```
1 #include <bits/stdc++.h>
2 using namespace std;
3 stack<int> n;
4 char ch;
5 int s,x,y;
6 int main()
7 {
8     while(ch!='@')
9     {
10         ch=getchar();
11         switch(ch)
12         {
13             case '+':x=n.top();n.pop();y=n.top();n.pop();n.push(x+y);break;
14             case '-':x=n.top();n.pop();y=n.top();n.pop();n.push(y-x);break;
15             case '*':x=n.top();n.pop();y=n.top();n.pop();n.push(x*y);break;
16             case '/':x=n.top();n.pop();y=n.top();n.pop();n.push(y/x);break;
17             case '.':n.push(s);s=0;break;
18             default :s=s*10+ch-'0';break;
19         }
20     }
21     printf("%d\n",n.top());
22     return 0;
23 }
```

选择

5. 输入

1.@2.+

输出是（ ）。

A. 1 B. 2

C. 3 D.没有输出

3分钟

阅读程序

```
1 #include <bits/stdc++.h>
2 using namespace std;
3 stack<int> n;
4 char ch;
5 int s,x,y;
6 int main()
7 {
8     while(ch!='@')
9     {
10         ch=getchar();
11         switch(ch)
12         {
13             case '+':x=n.top();n.pop();y=n.top();n.pop();n.push(x+y);break;
14             case '-':x=n.top();n.pop();y=n.top();n.pop();n.push(y-x);break;
15             case '*':x=n.top();n.pop();y=n.top();n.pop();n.push(x*y);break;
16             case '/':x=n.top();n.pop();y=n.top();n.pop();n.push(y/x);break;
17             case '.':n.push(s);s=0;break;
18             default :s=s*10+ch-'0';break;
19         }
20     }
21     printf("%d\n",n.top());
22     return 0;
23 }
```

选择

6. 输入中出现了除了+、-、*、/和@之外的符号，输出会（ ）。
- A. 程序运行不报错，没有输出
 - B. 程序运行报错
 - C. 程序运行不报错，正常输出，会实现额外符号的功能
 - D. 程序运行不报错，正常输出，但不会实现额外符号的功能

3分钟

阅读程序

```
1 #include <bits/stdc++.h>
2 using namespace std;
3 stack<int> n;
4 char ch;
5 int s,x,y;
6 int main()
7 {
8     while(ch!='@')
9     {
10         ch=getchar();
11         switch(ch)
12         {
13             case '+':x=n.top();n.pop();y=n.top();n.pop();n.push(x+y);break;
14             case '-':x=n.top();n.pop();y=n.top();n.pop();n.push(y-x);break;
15             case '*':x=n.top();n.pop();y=n.top();n.pop();n.push(x*y);break;
16             case '/':x=n.top();n.pop();y=n.top();n.pop();n.push(y/x);break;
17             case '.':n.push(s);s=0;break;
18             default :s=s*10+ch-'0';break;
19         }
20     }
21     printf("%d\n",n.top());
22     return 0;
23 }
```

判断

1. 程序使用了队列的数据结构。()

3分钟

阅读程序

```
1 #include <bits/stdc++.h>
2 using namespace std;
3 stack<int> n;
4 char ch;
5 int s,x,y;
6 int main()
7 {
8     while(ch!='@')
9     {
10         ch=getchar();
11         switch(ch)
12         {
13             case '+':x=n.top();n.pop();y=n.top();n.pop();n.push(x+y);break;
14             case '-':x=n.top();n.pop();y=n.top();n.pop();n.push(y-x);break;
15             case '*':x=n.top();n.pop();y=n.top();n.pop();n.push(x*y);break;
16             case '/':x=n.top();n.pop();y=n.top();n.pop();n.push(y/x);break;
17             case '.':n.push(s);s=0;break;
18             default :s=s*10+ch-'0';break;
19         }
20     }
21     printf("%d\n",n.top());
22     return 0;
23 }
```

判断

2. 符号越靠后面越早运算。
()

完善程序

快快编程
kkcoding.net

完善程序

1. (扫雷游戏)

扫雷游戏是一款十分经典的单机小游戏。在 n 行 m 列的雷区中有一些格子含有地雷（称之为地雷格），其他格子不含地雷（称之为非地雷格）。玩家翻开一个非地雷格时，该格将会出现一个数字——提示周围格子中有多少个是地雷格。游戏的目标是在不翻出任何地雷格的条件下，找出所有的非地雷格。

现在给出 n 行 m 列的雷区中的地雷分布，要求计算出每个非地雷格周围的地雷格数。
注：一个格子的周围格子包括其上、下、左、右、左上、右上、左下、右下八个方向上与之直接相邻的格子。

完善程序

输入：

第一行是用一个空格隔开的两个整数n和m，分别表示雷区的行数和列数。

接下来n行，每行m个字符，描述了雷区中的地雷分布情况。字符'*'表示相应格子是地雷格，字符'?'表示相应格子是非地雷格。相邻字符之间无分隔符。

输出：

输出文件包含n行，每行m个字符，描述整个雷区。用'*'表示地雷格，用周围的地雷个数表示非地雷格。相邻字符之间无分隔符。

样例：

输入：

3 3

*??

???

?*?

输出

*10

221

1*1

快快编程
kkcoding.net

完善程序

3分钟

```
#include<bits/stdc++.h>
using namespace std;
char a[101][101];
int b[101][101]={0};
int n,m,i,j;
int dx[8]={1,0,-1,0,-1,1,1,-1};
int dy[8]={0,1,0,-1,-1,-1,1,1};
void dfs(int x,int y)
{
    int nx,ny,k;
    for (k=0;k<8;k++)
    {
        nx=x+dx[k];
        ny=y+dy[k];
        if (nx>=1&&nx<=n&&ny>=1&&ny<=m)
            ---1---;
    }
}
int main()
{
    cin>>n>>m;
    for (i=1;i<=n;i++)
        for (j=1;j<=m;j++)
        {
            cin>>a[i][j];
            if (---2---)
                ---3---;
        }
    for (i=1;i<=n;i++)
    {
        for (j=1;j<=m;j++)
            if (---4---)
                cout<<a[i][j];
            else
                cout<<b[i][j];
            ---5---;
    }
    return 0;
}
```

- 1.程序太长就先提取关键部分
- 2.手算样例跟着程序走
- 3.理解题目中提供的思路

切记一点，填空已经变成选择题，结合题目！！

完善程序

```
void dfs(int x,int y)
{
    int nx,ny,k;
    for (k=0;k<8;k++)
    {
        nx=x+dx[k];
        ny=y+dy[k];
        if (nx>=1&&nx<=n&&ny>=1&&ny<=m)
            ---1---;
    }
}
```

选择

1. (1) 处应填 ()。

- A. b[nx][ny]-- B. b[nx][ny]++
C. b[nx][ny]=0 D. b[nx][ny]=1

完善程序

```
int main()
{
    cin>>n>>m;
    for (i=1;i<=n;i++)
        for (j=1;j<=m;j++)
        {
            cin>>a[i][j];
            if (---2---)
                ---3---;
        }
    for (i=1;i<=n;i++)
    {
        for (j=1;j<=m;j++)
            if (---4---)
                cout<<a[i][j];
            else
                cout<<b[i][j];
            ---5---;
        }
    return 0;
}
```

选择

2. (2) 处应填 ()。

A. `a[i][j]=='*`

B. `a[i][j]!='*`

C. `i==n`

D. `j==m`

完善程序

```
int main()
{
    cin>>n>>m;
    for (i=1;i<=n;i++)
        for (j=1;j<=m;j++)
        {
            cin>>a[i][j];
            if (---2---)
                ---3---;
        }
    for (i=1;i<=n;i++)
    {
        for (j=1;j<=m;j++)
            if (---4---)
                cout<<a[i][j];
            else
                cout<<b[i][j];
            ---5---;
        }
    return 0;
}
```

选择

3. (3) 处应填 ()。

- A. dfs(j,j) B. dfs(a[i][j]) C. dfs(i,j)
D. dfs(i,i)

完善程序

```
int main()
{
    cin>>n>>m;
    for (i=1;i<=n;i++)
        for (j=1;j<=m;j++)
        {
            cin>>a[i][j];
            if (---2---)
                ---3---;
        }
    for (i=1;i<=n;i++)
    {
        for (j=1;j<=m;j++)
            if (---4---)
                cout<<a[i][j];
            else
                cout<<b[i][j];
            ---5---;
        }
    return 0;
}
```

选择

4. (4) 处应填 ()。

- A. $b[i][j] \neq 0$ B. $b[i][j] == 0$
C. $a[i][j] \neq '*'$ D. $a[i][j] == '*'$

完善程序

```
int main()
{
    cin>>n>>m;
    for (i=1;i<=n;i++)
        for (j=1;j<=m;j++)
        {
            cin>>a[i][j];
            if (---2---)
                ---3---;
        }
    for (i=1;i<=n;i++)
    {
        for (j=1;j<=m;j++)
            if (---4---)
                cout<<a[i][j];
            else
                cout<<b[i][j];
            ---5---;
        }
    return 0;
}
```

选择

5. (5) 处应填 ()。
- A. break B. cout<<endl
C. cout<<' ' D. continue

完善程序

快快编程
kkcoding.net

完善程序

2. (回文数)

若一个数(首位不为0)从左向右读与从右向左读都一样，我们就将其称为回文数。

例如：给定一个十进制数56，将56+65，得到121是一个回文数。

又比如对于十进制数87

1. $87+78=165$

2. $165+561=726$

3. $726+627=1353$

4. $1353+3531=4884$

在这里一步是指进行了一次N进制的加法，上面样例最少使用4步的到回文数4884。

写一个程序，给定一个N($2 \leq N \leq 10$ 或 $N=16$)进制数M(100位以内)，求最少经过几步可以得到回文数。如果在30步以内(包含30步)不可能得到回文数，则输出Impossible!。

手算样例

输入格式

两行，分别是N， M

输出格式

如果能在30步以内的到回文数输出格式形如 STEP=ans，其中 ans 为最少得到回文数的步数。

否则输出 Impossible!。

样例

输入：

10

87

输出：

STEP=4

快快编程
kkcoding.net

完善程序

```
int main() {  
    cin >> n >> m;  
    for (int i=0; i<=30; i++){  
        if(hw(m)) {  
            printf("STEP=%d", i);  
            return 0;  
        }  
        else m = add(n, m);  
    }  
    printf("---5---");  
    return 0;  
}
```

- 1.程序基本组成部分：输入，输出
- 2.带上手算样例，边看边写下遇到的变量
- 3.分析要结合选择题选项

完善程序

```
int main() {  
    cin >> n >> m;  
    for (int i=0; i<=30; i++){  
        if(hw(m)) {  
            printf("STEP=%d", i);  
            return 0;  
        }  
        else m = add(n, m);  
    }  
    printf("---5---");  
    return 0;  
}
```

选择

5. (5) 处应填 ()。

A. Impossible!

B. Impossible

C. "STEP=%d", N

D. "STEP=%d", i

完善程序

```
bool hw(string a) {  
    string s = a;  
    reverse(s.begin(), s.end()); //reverse函数的作用是反转字符串 s  
    return ---1---;  
}
```

```
int main() {  
    cin >> n >> m;  
    for (int i=0; i<=30; i++){  
        if(hw(m)) {  
            printf("STEP=%d", i);  
            return 0;  
        }  
        else m = add(n, m);  
    }  
    printf("---5---");  
    return 0;  
}
```

选择

1. (1) 处应填 ()。

A. 0

B. s == a

C. 1

D. s!= a

完善程序

```
string add(int k, string b) {  
    string a = b;  
    reverse(a.begin(), a.end());  
    int numa[maxN], numb[maxN], numc[maxN];  
    int len = a.length(), lenc = 1;  
    string ans;  
    for (int i=0; i<len; i++) {  
        if (isdigit(a[i])) numa[len-i] = a[i] - '0'; //isdigit函数判断是不是数字  
  
        else numa[len-i] = a[i] - 'A' + 10;  
        if (isdigit(b[i])) numb[len-i] = b[i] - '0';  
        else numb[len-i] = b[i] - 'A' + 10;  
    }  
    int x = 0;  
    while (lenc <= len) {  
        numc[lenc] = ---2---;  
        x = numc[lenc] / k;  
        ---3---;  
        lenc++;  
    }  
    numc[lenc] = x;  
    while (numc[lenc] == 0) lenc--;  
    for (int i=lenc; i>=1; i--) ans +=---4---;  
    return ans;  
}
```

选择

2. (2) 处应填 ()。

- A. numa[lenc] + numb[lenc]
- B. numa[lenc] + x
- C. numa[lenc] + numb[lenc] + x
- D. numb[lenc] + x

完善程序

```
string add(int k, string b) {  
    string a = b;  
    reverse(a.begin(), a.end());  
    int numa[maxN], numb[maxN], numc[maxN];  
    int len = a.length(), lenc = 1;  
    string ans;  
    for (int i=0; i<len; i++) {  
        if (isdigit(a[i])) numa[len-i] = a[i] - '0'; //isdigit函数判断是不是数字  
  
        else numa[len-i] = a[i] - 'A' + 10;  
        if (isdigit(b[i])) numb[len-i] = b[i] - '0';  
        else numb[len-i] = b[i] - 'A' + 10;  
    }  
    int x = 0;  
    while (lenc <= len) {  
        numc[lenc] = ---2---;  
        x = numc[lenc] / k;  
        ---3---;  
        lenc++;  
    }  
    numc[lenc] = x;  
    while (numc[lenc] == 0) lenc--;  
    for (int i=lenc; i>=1; i--) ans +=---4---;  
    return ans;  
}
```

选择

3. (3) 处应填 () 。

A. numc[lenc] %= k

B. numc[lenc] %= 10

C. numc[lenc]--

D. numc[lenc+1]--

完善程序

```
string add(int k, string b) {  
    string a = b;  
    reverse(a.begin(), a.end());  
    int numa[maxN], numb[maxN], numc[maxN];  
    int len = a.length(), lenc = 1;  
    string ans;  
    for (int i=0; i<len; i++) {  
        if (isdigit(a[i])) numa[len-i] = a[i] - '0'; //isdigit函  
数判断是不是数字  
  
        else numa[len-i] = a[i] - 'A' + 10;  
        if (isdigit(b[i])) numb[len-i] = b[i] - '0';  
        else numb[len-i] = b[i] - 'A' + 10;  
    }  
    int x = 0;  
    while (lenc <= len) {  
        numc[lenc] = ---2---;  
        x = numc[lenc] / k;  
        ---3---;  
        lenc++;  
    }  
    numc[lenc] = x;  
    while (numc[lenc] == 0) lenc--;  
    for (int i=lenc; i>=1; i--) ans +=---4---;  
    return ans;  
}
```

选择

4. (4) 处应填 () 。

- A. numb[i]
- B. numc[i]
- C. numa[i]
- D. sixt[numc[i]]