提高级 CSP-S 第 3 套初赛模拟试题

一、单项选择题(共 15 题,每题 2 分,共计 30 分;每题有且仅有一个正确选项)

1. 假设有以下定义:int a[5]={1,2,3,4,5},i=3,*p=a,*q=a;则不能正确执行的语句是(　　)。
   A. i=*p+*q;　　B. a=i;
   C. *p=*(a+i);　　D. i=*p**(q+2);

2. 下列不属于 CPU 的有(　　)。
   A. 海思麒麟 990　　B. Intel 酷睿 i7　　C. 影驰 RTX2070　　D. AMD Ryzen 7

3. $(2019)_{10}+(2020)_8$ 的结果是(　　)。
   A. $(3049)_{10}$　　B. $(BF3)_{16}$
   C. $(101111110001)_2$　　D. $(5765)_8$

4. 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

5. 若有定义 char x[ ]="12345"; char y[ ]={'1','2','3','4','5'};则(　　)。
   A. x 数组与 y 数组的所占内存空间相同　　B. x 数组比 y 数组占的内存空间大
   C. x 数组比 y 数组所占内存空间小　　D. x 数组等价于 y 数组

6. 公共汽车起点站于每小时的 10 分,30 分,55 分发车,该顾客不知发车时间,在每小时内的任一时刻随机到达车站,如果乘客到车站的时刻为发车时间,乘客不能坐上此时发车的公共汽车,则乘客候车时间的数学期望(准确到秒)是(　　)。
   A. 8 分 40 秒　　B. 15 分 20 秒　　C. 22 分 30 秒　　D. 10 分 25 秒

7. 设要将序列<Q,H,C,Y,P,A,M,S,R,D,F,X>中的关键码按字母的升序重新排列,则(　　)是以第一个元素为分界元素的快速排序一趟扫描的结果。
   A. F,H,C,D,P,A,M,Q,R,S,Y,X　　B. P,A,C,S,Q,D,F,X,R,H,M,Y
   C. A,D,C,R,F,Q,M,S,Y,P,H,X　　D. H,C,Q,P,A,M,S,R,D,F,X,Y

8. 设 G 是有 n 个结点、m 条边($n\leqslant m$)的连通图,必须删去 G 的(　　)条边,才能使得 G 变成一棵树。
   A. m-n+1　　B. m-n　　C. m+n+1　　D. n-m+1

9. 将 2 个红球,1 个蓝球,1 个白球放到 10 个编号不同的盒子中去,每个盒子最多放一个球,有多少种放法(　　)。
   A. 5040　　B. 2520　　C. 1260　　D. 420

10. 一个家具公司生产桌子和椅子。现在有 113 个单位的木材。每张桌子要使用 20 个单位的木材,售价是 30 元;每张椅子要使用 16 个单位的木材,售价是 20 元。使用已有的木材生产桌椅(不一定要把木材用光),最多可以卖(　　)元钱。
    A. 140　　B. 150　　C. 160　　D. 170

11. 给出 4 种排序:插入排序、冒泡排序、选择排序、快速排序。这 4 种排序的时间代价分别是(　　)。
    A. $O(n^2)$、$O(n^2)$、$O(n^2)$、$O(n\log n)$　　B. $O(n^2)$、$O(n^2)$、$O(n^2)$、$O(\log n)$
    C. $O(n\log n)$、$O(n^2)$、$O(n^2)$、$O(n\log n)$　　D. $O(n\log n)$、$O(n^2)$、$O(n\log n)$、$O(n\log n)$

12. 以下数据结构中,哪一个不是线性结构?(　　)。
    A. 广义表　　B. 二叉树　　C. 队列　　D. 栈

13. 以下最短路算法中不能处理带有负权值的算法的是(　　)。
    A. Dijkstra 算法　　B. Floyd 算法
    C. Bellman-Ford 算法　　D. SPFA 算法

14. 栈 S 最多能容纳 4 个元素。现有 6 个元素按 1,2,3,4,5,6 的顺序进栈,问下列哪一个序列是可能的出栈序列?(　　)。

A. 5, 4, 3, 2, 1, 6　　B. 3, 2, 5, 4, 1, 6

C. 2, 3, 5, 6, 1, 4　　D. 1, 4, 6, 5, 2, 3

15. 平面上有三条平行直线,每条直线上分别有 7,5,6 个点,且不同直线上三个点都不在同一条直线上。问用这些点为顶点,能组成(　　)个不同四边形。

A. 18　　B. 210　　C. 2250　　D. 4500

二、阅读程序(程序输入不超过数组或字符串定义的范围;判断题正确的填√,错误的填×;除特殊说明外,判断题每题 1.5 分,选择题每题 4 分,共计 40 分)

1.

```
01 #include<cstdio>
02 using namespace std;
03 int findvall(int n)
04 {
05         int f;
06         if(n==0) return 1;
07         else
08         {
09                 f=findvall(n/2);
10                 return (n*f);
11         }
12 }
13 int main()
14 {
15         int n;
16         scanf("%d",&n);
17         printf("%d\n",findvall(n));
18         return 0;
19 }
```

●判断题

(1)第 06 行输出 if(n==0)改成 if(n==1)时,对于输入的正整数 n,输出结果不会改变。(　　)

(2)对于输入的正整数程序输出的值小于等于 n。(　　)

(3)如果输入的 n 是负数的话,该程序会出现死循环,所以该程序不能求解 n 是负数的情况。(　　)

(4)如果多次运行该程序,并且输入的 n 是单调递增的正整数,那么每次输出的结果也是一个严格单调递增的数列。(　　)

●选择题

(5)若两次输入 n 的值相差 1,但输出的结果却是一个正数,一个负数,那么两次输入的 n 可能是下面四组中的(　　)。

A. 不可能　　B. -6,-7　　C. -15,-16　　D. -23,-24

(6)此程序的时间复杂度是(　　)。

A. $O(n^2)$　　B. $O(\log n)$　　C. $O(n)$　　D. $O(n \log n)$

2. 输入一串由小写字母组成的字符串,根据程序判断或选择正确的答案。

```
01 #include<cstdio>
02 #include<cstring>
```

```
03 using namespace std;
04 int main() {
05     char str[60];
06     int len,i,j,chr[26];
07     char mmin=' z';
08     scanf("%s",str);
09     len=strlen(str);
10     for (i=len-1;i>=1;i--)
11         if (str[i-1]<str[i]) break;
12     if (i==0) {
13         printf("No result!\n");
14         return 0;
15     }
16     for (j=0;j<i-1;j++) putchar(str[j]);
17     memset(chr,0,sizeof(chr));
18     for (j=i;j<len;j++) {
19         if (str[j]>str[i-1] && str[j]<mmin)
20             mmin=str[j];
21         chr[str[j]-' a']++;
22     }
23     chr[mmin-' a']--;
24     chr[str[i-1]-' a']++;
25     putchar(mmin);
26     for (i=0;i<26;i++)
27         for (j=0;j<chr[i];j++)
28             putchar(i+' a');
29     putchar('\n');
30     return 0;
31 }
```

●判断题

(1)输入的字符串长度应该在[1,59]的范围内。(　　)

(2)如果输入的字符数组所有字符都是从大到小的,那么会输出“No result!”。(　　)

(3)第 25 行输出的值为输入字符串里的 ASCII 最小的那个字符。(　　)

(4)第 26 行到第 28 行是把 chr[ ]数组中存在的对应字符按照从小到大输出,即把剩下未输出的字符按照从小到大输出。(　　)

●选择题

(5)如果输入的是 abcdzdcba,则第 16 行输出的是(　　)。

A. abc　　B. abcd　　C. abcdz　　D. abcdzd

(6)如果程序的输出结果是“ffghhggh”,则输入有可能是(　　)。

A. ffghhghg　　B. ffghhhgg　　C. ffghghhg　　D. ffghghgh

3. 本题是一款模拟贪吃蛇程序,游戏是在一个 a * a 的网格上进行的。其中输入第一行一个整数 a。第二行两个整数 n 和 m。接下来是 n 行,每行第一个数为 opt,表示操作编号。接下来的输入的变量与操作编号对应,输出:即第 m 秒过后的地图,蛇所在的位置输出“o”,其余位置输出“.”,以换行结尾。

```
01 #include<bits/stdc++.h>
02 #include<windows.h>
```

```
using namespace std;
int a,mp[101][101];
int t[100003];
int y[100003];
int cnt;
int len=2,dir=3,die=0;
const int dx[5]={0,0,-1,0,1};
const int dy[5]={0,-1,0,1,0};
int nx=0,ny=1;
int px=1,py=2;
int check(int x,int yy)
{
    if(x<1 ||x>a ||yy<1 ||yy>a)
        return 1;
    if(cnt+1-mp[x][yy]<len)
        return 1;
    return 0;
}
void work()
{
    if(die) return;
    px+=nx;
    py+=ny;
    die=check(px,py);
    if(die) return;
    mp[px][py]=++cnt;
}
void show()
{
    for(int i=1;i<=a;++i)
    {
        for(int j=1;j<=a;++j)
        {
            if(mp[i][j] !=0 && mp[i][j]>=cnt-len+1)
                putchar('o');
            else
                putchar('.');
        }
        puts("");
    }
}
int main()
{
    mp[1][1]=++cnt;
    mp[1][2]=++cnt;
    int n,m,op,xx;
```

```
49      char s[3];
50      scanf("%d",&a);
51      scanf("%d%d",&n,&m);
52      while(n--)
53      {
54          scanf("%d%d",&op,&xx);
55          if(op==1)
56          {
57              t[xx]=1;
58              scanf("%s",s);
59              if(s[0]=='L')
60                  y[xx]=1;
61              else if(s[0]=='U')
62                  y[xx]=2;
63              else if(s[0]=='R')
64                  y[xx]=3;
65              else
66                  y[xx]=4;
67          }
68          else
69          {
70              t[xx]=2;
71          }
72      }
73      for(int tm=1;tm<=m;++tm)
74      {
75          if(t[tm]==1)
76          {
77              if(y[tm]%2 !=dir%2)
78              {
79                  dir=y[tm];
80                  nx=dx[y[tm]];
81                  ny=dy[y[tm]];
82              }
83          }
84          else if(t[tm]==2)
85          {
86              ++len;
87          }
88          work();
89          if(die)
90          {
91              break;
92          }
93      }
94      show();
```

```
95      return 0;
96  }
```

●判断题：

(1)(2分)由程序代码可知,贪吃蛇的初始长度为2,蛇头和蛇尾分别在坐标(1,2)、(1,1)处。(　　)

(2)(2分) check 函数是用来检测蛇是否吃到果实的。(　　)

(3)(2分)第54行及第58行输入1 x y 表示在第x秒按下了y键,y为LURD中的一种,分别表示按下了左、上、右、下四种按钮。(　　)

(4)(2分) 当输入样例如下所示时：

```
10
10 20
2 1
2 2
2 3
2 4
2 5
1 6 R
1 7 D
1 8 L
1 9 U
```

最终程序的运行结果所代表的含义可表示为贪吃蛇在第9秒过后就死亡了,因此最后贪吃蛇保持的是死亡前(第7秒过后)的位置。(　　)

●选择题：

(5)若输入地图边长为x,共n次操作(x>n),则该程序时间复杂度为(　　)。

A. $O(x^2)$　　B. $O(n^2)$　　C. $O(n^2*x)$　　D. $O(x^2*n)$

三、完善程序(单选题,每题3分,共计30分)

1.(APP点餐)小D在外玩,回来时想在APP上点KFC,然后回宿舍。他想选择一家KFC,取了食品后尽快回宿舍,忽略取餐时间。地图可看作是n*m的网格,其中有一些不可通过的障碍,小D、KFC、宿舍均可以看做是道路,可以通过,他可穿过多个KFC。小D可以选择上下左右四个方向移动,一次移动算一步。求他取到餐并回到宿舍的最小步数。

输入第一行有两个数n,m。

接下来n行,每行包含m个字符,表示地图。
"S"表示小D初始位置,
"E"表示宿舍位置,
"#"表示障碍物,
"."表示道路,
"K"表示KFC。

下面的程序已采用宽度优先搜索的算法完成这个问题,试补全程序。

```
01  #include<bits/stdc++.h>
02  #define fi first
03  #define se second
04  using namespace std;
05  const int MAXN=1e3+10;
```

```
const int INF=0x3f3f3f3f;
typedef pair<int,int>P;
char s[MAXN][MAXN];
int n,m;
int dir[2][4]={{1,-1,0,0},{0,0,1,-1}};
int dis[2][MAXN][MAXN];

void bfs(int p,int a,int b){
  memset(dis[p],INF,sizeof(dis[p]));
  dis[p][a][b]=0;
  queue<pair<P,int>>q;
  q.push({{a,b},0});
  while(!q.empty()){
    int x=q.front().fi.fi,
      y=q.front().fi.se,
      d=q.front().se;
    q.pop();
    for(int i=0;i<4;i++){
      int dx=x+dir[0][i],
        dy=y+dir[1][i];
      if(dx<1||dy<1||dx>n||dy>m||s[dx][dy]=='#')
        continue;
      if(    ①    ){
        ②    ;
        ③    ;
      }
    }
  }
}

int main(void)
{
  while(scanf("%d%d",&n,&m)!=EOF){
    int ans=INF;
    scanf("%d%d",&n,&m);
    for(int i=1;i<=n;i++)
      scanf("%s",s[i]+1);
    for(int i=1;i<=n;i++)
      for(int j=1;j<=m;j++)
        if(s[i][j]=='S')
          bfs(0,i,j);
      else if(s[i][j]=='E')
            ④    ;
    for(int i=1;i<=n;i++)
      for(int j=1;j<=m;j++)
        if(s[i][j]=='K')
```

```
52          ans=      ⑤      ;
53        if(ans==INF)
54          ans=-1;
55        printf("%d\n",ans);
56      }
57      return 0;
58  }
```

(1)①处应填(　　)。

A. dis[p][dx][dy]>d　　B. dis[p][dx][dy]>d+1

C. dis[p][dx][dy]<d　　D. dis[p][dx][dy]<d+1

(2)②处应填(　　)。

A. dis[p][dx][dy]=d　　B. dis[p][dx][dy]=d-1

C. dis[p][dx][dy]=d+1　　D. dis[p][dx][dy]=1

(3)③处应填(　　)。

A. q.push({{dx,dy},d+1})　　B. q.push({{dx,dy},d})

C. q.push({{dx,dy},d-1})　　D. q.push({{dx,dy},1})

(4)④处应填(　　)。

A. bfs(0,j,i)　　B. bfs(1,j,i)

C. bfs(0,i,j)　　D. bfs(1,i,j)

(5)⑤处应填(　　)。

A. min(ans,dis[0][i][j]+dis[1][i][j])

B. min(ans,dis[0][j][i]+dis[1][j][i])

C. min(ans,dis[0][i][j])

D. min(ans,dis[1][i][j])

2.(尺取法求区间个数)给n个非负整数a[1],a[2],…,a[n],求区间和小于或等于k的区间个数,即求使SUM=a[L]+a[L+1]+…+a[R-1]+a[R]<=k的区间[L,R]的个数(1≤L≤R≤n),但由于对内存和复杂度有要求,本题已经用尺取法写好部分代码,请补全程序。

输入第一行两个整数n,k(1≤n≤1000000,0≤k≤10000000000000000)。

第二行为n个数,表示a[1]~a[n]的值(0≤a[i]≤1000000000)。

```
01  #include<bits/stdc++.h>
02  #define ll long long
03  using namespace std;
04  const int mx=1e6+10;
05  int n,a[mx];
06  ll k,sum,ans;
07  int main()
08  {
09    scanf("%d%lld",&n,&k);
10    for (int i=1;i<=n;++i)
11    {
12        scanf("%d",&a[i]);
13    }
14    int r=0;
15    for (int i=1;i<=n;++i)
16    {
```

```
17        while (r<n)
18        {
19            if ( ____①____ )        ____②____;
20            else break;
21        }
22        ____③____;
23        if (i<=r)  ____④____;
24        else  ____⑤____;
25    }
26    printf("%lld\n",ans);
27  }
```

(1)①处应填(　　)。

A. sum+a[r+1]<=k　　B. sum+a[r]<=k

C. sum+a[r+1]<k　　D. sum+a[r]<k

(2)②处应填(　　)。

A. sum+=a[r]　　B. sum+=a[++r]

C. sum+=a[r++]　　D. sum+=a[r+1]

(3)③处应填(　　)。

A. ans+=r-i+1　　B. ans+=r-i

C. ans+=r-i-1　　D. ans+=n-i+1

(4)④处应填(　　)。

A. sum+=a[i]　　B. sum=a[r]

C. sum=a[i]　　D. sum-=a[i]

(5)⑤处应填(　　)。

A. r=++i　　B. r=i--

C. r=i++　　D. r=i