

BCSP 初中组基础知识测评样卷

(C++语言 满分: 100 分 考试时间: 120 分钟)

一. 单项选择题 (每题只有一个正确选项, 每题 2 分, 共 30 分)

1. 世界上第一台电子计算机诞生于 ()

A. 1945 年 B. 1949 年 C. 1942 年 D. 1948 年 E. 1946 年 F. 1951 年

2. 盒子里面有 5 个黄球, 5 个蓝球, 5 个红球, 这 15 个球形状完全相同, 现在要从盒子中依次取出 4 个球(每次取完会放回), 并且要满足取出的这 4 个球中至少有一个红球, 方案数是 ()

A. 120 B. 32 C. 56 D. 92 E. 108 F. 65

3. 下列关于 C++ 类的成员函数的表述中, 哪一个是正确的? ()

- A. 构造函数可以有返回值, 且可以被重载。
- B. 纯虚函数必须在派生类中实现, 使得基类成为抽象类。
- C. 静态成员函数只能通过对象调用, 而不能通过类名调用。
- D. 析构函数可以重载, 以处理不同的清理需求。
- E. 构造函数必须定义, 不能缺省。
- F. 成员函数一定是内联函数。

4. () 的每一次操作, 其均摊复杂度和最坏复杂度是同阶的。

- A. `std::vector` 的 插入
- B. 斜堆的合并
- C. 二分的单调队列的插入
- D. `splay` 实现的平衡树的插入
- E. 栈的 `push` 操作
- F. 哈希表的查找操作

5. 若元素 a, b, c, d, e, f, g 依次进栈, 栈的大小为 3, 以下哪一种出栈顺序是不可能的? ()

- A. c, b, a, d, e, f, g
- B. b, c, d, a, f, e, g
- C. b, d, c, e, g, f, a
- D. a, d, e, b, c, g, f
- E. c, d, e, f, g, b, a
- F. c, b, d, e, f, g, a

6. 关于最小生成树，下列说法正确的是（ ）
- A. 最小生成树唯一，当且仅当边权两两不相等。
- B. 使用完全二叉堆优化的 Prim 算法，复杂度为 $\Theta(|E| + |V| \log |V|)$ 。
- C. 使用 Borůvka 算法，并查集使用最佳的优化，复杂度为 $\Theta((|V| + |E|) \alpha(|V|))$ 。
- D. 先用 Prim 算法加入若干点和边，将这些点和边视为一个点，再对未被加入最小生成树的边用 Kruskal 算法，能求出最小生成树。
- E. 所有权值最小的边一定会出现在所有的最小生成树中
- F. 使用普里姆（Prim）算法从不同顶点开始得到的最小生成树一定相同
7. 对于一个二叉树，其前序遍历为 A, B, D, E, C, F, G，其后序遍历为 D, E, B, G, F, C, A，其可能对应的二叉树的数量是多少？（ ）
- A. 1 B. 5 C. 3 D. 6 E. 7 F. 4
8. 当 $n \leq 10^6$ 时，算法需要执行的指令数量为 $n \log n$ 条，该算法的时间复杂度是（ ）。
- A. $O(n)$ B. $O(n \log n)$ C. $\omega(n \log n)$ D. 无法确定
- E. $\Omega(n \log n)$ F. $O(\log n)$
9. 已知某表达式的后缀表达式是 $A B C - \times D E F + + \div$ ，它的前缀表达式是（ ）。
- A. $\div \times A - B C + + D E F$ B. $\div \times A B - C + + D E F$
- C. $\times A \div - B C + + D E F$ D. $\times A \div - B C + D + E F$
- E. $\div \times A - B C + D + E F$ F. $\times A \div - C B + D + E F$
10. 4 个点的无向满图，节点互不相同，有（ ）种不同的生成树。
- A. 6 B. 12 C. 14 D. 16
- E. 18 F. 24
11. 以下哪一个选项与 $(12.34)_8$ 的值相等？（ ）
- A. $(10.425)_{10}$
- B. $(1010.0111)_2$
- C. $(A.22)_{16}$
- D. $(1001.1010)_2$

E. $(1011.1000)_2$

F. $(9.192)_{16}$

12. 某算法的复杂度 $T(n)$ 满足:

$$T(n) = \begin{cases} 1 & 0 \leq n \leq 1 \\ 4T\left(\frac{n}{2}\right) + n \log n & n > 1 \end{cases}$$

则 $T(n)$ 的渐进表示为 ()。

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

E. $\Theta(n)$ F. $\Theta(n^2)$

13. 液晶时钟由 6 个数位构成, 每个数位有 7 块液晶。已知时间按 24 小时制计, 每个数字只有一种合法的显示形式。假设每块液晶各自



独立等概率点亮或熄灭, 表示分隔符的冒号忽略不计, 时钟显示的是合法时间的概率是 ()。

A. $\frac{54}{625}$ B. $\frac{675}{2^{36}}$ C. $\frac{1875}{2^{35}}$ D. $\frac{9375}{2^{37}}$

E. $\frac{128}{1173}$ F. $\frac{675}{2^{35}}$

14. 单向链表中的每个节点用 `next` 记录了后继节点的地址, 现在要删除指针 `p` 指向的节点的后两个点 (保证后两个节点都存在), 可以使用语句 ()

A. `p->next=p->next->next`

B. `p->next=p->next->next->next;`

C. `p->next->next=p->next->next->next;`

D. `p=p->next,p=p->next,p=p->next;`

E. `p->next->next = p;`

F. `p=p->next->next,p->next=p;`

15. 对于以下 C++ 代码, 其输出为 ()。

01 `#include <iostream>`

02 `int main() {`

03 `char x = -064;`

```

04 unsigned char y = 0xcc;
05 x <<= 4;
06 y >>= 1;
07 std::cout << int(x) << ' ' << int(y);
08 }

```

A. -64 102

B. 0 102

C. -64 110

D. 包含未定义行为，无法确定输出

E. -064

F. 128

二、程序阅读理解题（共 3 大题。程序输入不超过数组或字符串定义的范围，除特殊说明外，判断题 1.5 分，选择题 3 分，共计 40 分）

(1) (14.5 分)

```

01  #include <stdio>
02  void swap(int* a, int* b){
03      int t = *a;
04      *a = *b;
05      *b = t;
06  }
07  int f(int a, int b){
08      while(b){
09          a %= b;
10          swap(a, b);
11      }
12      return a;
13  }
14  int main(){
15      int a, b, n;
16      scanf("%d%d%d", &a, &b, &n);
17      for(int i = 1; i <= n; i++){
18          a += b;
19          if(a % 2 == 0)
20              swap(a, b);
21      }
22      printf("%d\n", f(a, b));

```

```

23     return 0;
24 }

```

● 判断题

16. 第 10、20 行的 `swap(a, b)` 应当改为 `swap(&a, &b)`, 否则程序一定会出错。
17. 在第 7、8 行之间应当加入 `if(b == 0) return a;` 否则程序可能出现除以整数零的运行时错误。
18. 输入 72 108 0, 输出是 36。
19. 输入 72 108 3, 输出是 72。
20. `f` 函数的作用是计算 `a` 和 `b` 的最大公约数。

● 选择题

21. 输入 375 225 30, 程序的运行输出为 ()。

A. 1275 B. 225 C. 600 D. 17625 E. 75 F. 128

22. 2~6 行的等价修改方式为 ()

A.	<pre> void swap(int* a, int* b){ int *t = *a; *a = *b; *b = *t; } </pre>	B.	<pre> void swap(int& a, int& b){ int t = a; a = b; b = t; } </pre>
C.	<pre> void swap(int* const a, int* const b){ int t = *a; *a = *b; *b = t; } </pre>	D.	<pre> void swap(const int* a, const int* b){ int t = *a; *a = *b; *b = t; } </pre>
E	<pre> void swap(int a, int b){ int *t = *a; *a = *b; *b = *t; } </pre>	F	<pre> void swap(int& a, int& b){ int t = &a; a = b; b = t; } </pre>

(2) (12 分)

```

01.#include <bits/stdc++.h>
02.using namespace std;
03.const int MAXN = 405;
04.int main() {
05.    int n, m;
06.    cin >> n >> m;
07.    bool a[MAXN][MAXN];
08.    int sRow[MAXN][MAXN], sCol[MAXN][MAXN];
09.    for (int i = 1; i <= n; ++i) {
10.        string str;
11.        cin >> str;
12.        for (int j = 1; j <= m; ++j) {
13.            a[i][j] = str[j - 1] == '1';
14.            sRow[i][j] = a[i][j] + sRow[i][j - 1];
15.            sCol[i][j] = a[i][j] + sCol[i - 1][j];
16.        }
17.    }
18.    int ans = INT_MAX;
19.    for (int i = 1; i <= n; ++i) {
20.        for (int j = i + 4; j <= n; ++j) {
21.            int cnt = 0;
22.            cnt += (j - i - 1) - (sCol[j - 1][1] - sCol[i][1]);
23.            cnt += (!a[i][2] + !a[i][3]) + (!a[j][2] + !a[j][3]);
24.            cnt += (sCol[j - 1][2] - sCol[i][2]) +
25.                (sCol[j - 1][3] - sCol[i][3]);
26.            ans = min(ans,
27.                cnt + (j - i - 1) - (sCol[j - 1][4] - sCol[i][4]));
28.            for (int r = 5; r <= m; ++r) {
29.                cnt += !a[i][r - 1] + !a[j][r - 1] +
30.                    (sCol[j - 1][r - 1] - sCol[i][r - 1]);
31.                int newCnt = 0;
32.                newCnt +=
33.                    (j - i - 1) - (sCol[j - 1][r - 3] - sCol[i][r - 3]);
34.                newCnt += (!a[i][r - 1] + !a[i][r - 2]) +
35.                    (!a[j][r - 1] + !a[j][r - 2]);
36.                newCnt += (sCol[j - 1][r - 1] - sCol[i][r - 1]) +
37.                    (sCol[j - 1][r - 2] - sCol[i][r - 2]);
38.                cnt = min(cnt, newCnt);
39.                ans = min(ans, cnt + (j - i - 1) -
40.                    (sCol[j - 1][r] - sCol[i][r]));
41.            }
42.        }
43.    }
44.    cout << ans << endl;

```

45. `return 0;`

46. }

输入保证 n 小于等于 400, 大于等于 5, m 小于等于 400, 大于等于 4。第 11 行代码中的 `str` 为只包含 0 或 1 的字符串。

● 判断题

23. 对于第 30 行代码, 将其更换为 `newCnt += 4 - a[i][r - 1] - a[i][r - 2] - a[j][r - 1] - a[j][r - 2];`, 不会对程序运行结果产生任何影响。

24. 设一次操作可以将一个 0 转换为 1 或一个 1 转换为 0, 这段程序希望计算给定的 01 矩阵中, 使其存在一个大小至少为 5 行 4 列的矩形区域, 其边界 (包含四个角) 均为 1, 其他区域均为 0 的需要的最少操作次数。 () 57. 输出的字符串长度一定会大于 11 或 12。

25. 对于第 21 行至 26 行代码, 将其更换为 `int cnt = 400 * 400; for (int r = 4; r <= m; ++r) {`, 不会对程序运行结果产生任何影响。

26. 代码中使用了预处理矩阵的行和列累加和 (`sRow` 和 `sCol`) 来加速计算, 使得在选定矩形区域内的计算复杂度大幅降低。

● 选择题

27. 当输入为

6 4

1000

0000

0110

0000

0001

1001

时, 程序的输出为 ()

A. 11 B. 12 C. 13 D. 14 E. 15 F. 16

28. 当输入为

9 9

001010001

101110100

000010011

100000001

101010101

110001111

000001111

111100000

000110000

时，程序的输出为（）

A. 5 B. 6 C. 7 D. 8 E. 9 F. 10

(3) (13.5 分)

```
01 #include <bits/stdc++.h>
02 using namespace std;
03
04 #define endl "\n"
05 typedef long long ll;
06 typedef pair<int, int> PII;
07 typedef pair<PII, int> PIII;
08 const int inf = 0x3f3f3f3f;
09 const ll inff = 0x3f3f3f3f3f3f3f3f;
10
11 const int N = 3e5 + 10;
12 int l[N], r[N];
13 int n;
14 string s;
15
16 void dfs(int i, int cnt, int &res) {
17     if (!l[i] && !r[i]) res = min(res, cnt);
18
19     if (l[i]) {
20         dfs(l[i], cnt + (s[i - 1] != 'L'), res);
21     }
22     if (r[i]) {
23         dfs(r[i], cnt + (s[i - 1] != 'R'), res);
24     }
25 }
26
27 void solve() {
28     cin >> n >> s;
29
30     for (int i = 1; i <= n; ++i) cin >> l[i] >> r[i];
31
32     int ans = inf;
```



```

33     dfs(1, 0, ans);
34     cout << ans << endl;
35 }
36
37 int main() {
38     ios::sync_with_stdio(false);
39     cin.tie(nullptr);
40
41     int t; cin >> t;
42     while (t--) solve();
43
44     return 0;
45 }

```

(约定每次输入的 `str` 均由 2 个大写或小写字母拼接而成, 如 "aa", "Xt", "RE")

●判断题

29. 可以通过逐步移动或修改顶点上的字母, 最终到达叶子节点。
30. 如果在当前顶点上遇到字母 "R", 会尝试移动到父节点。
31. 每次修改字母时, 都会增加到达叶子节点的最短路径长度。
32. 如果将 `s` 中的任意一个 'L' 或 'R' 修改为 'U', 则输出答案一定变大。

●选择题

33. 如果当前顶点的字母是 "L", 接下来会 ()
 - A. 移动到父顶点
 - B. 移动到左子顶点
 - C. 移动到右子顶点
 - D. 停留在当前顶点不动
 - E. 移动到左子顶点的左子顶点
 - F. 移动到右子顶点的左子顶点
34. 以下输入对应的输出是 ()

1

7

LLRRRLU

5 2

3 6

0 0

7 0

4 0

0 0

0 0

A. 0 B. 1 C. 2 D. 3 E. 4 F. 5

三、程序完善题（共 2 大题，每个选择题 3 分，共计 30 分）

1. 题目描述：有一个 n 个点 m 条边的有向图，每个点都有一个权值，现在需要求出每个点能到达的所有点中最大的权值是多少呢？数据范围： $1 \leq n \leq 10^6$ ， $1 \leq m \leq 10^6$ ，时间限制 1s，试着补全程序

```
01 #include<bits/stdc++.h>
02
03 using namespace std;
04
05 #define N 1000006
06
07 int n,m;
08 int a[N],vis[N];
09 vector<int >edge[N];
10 queue<int >Q;
11 struct node {
12     int id;
13     bool friend operator < (node A,node B){
14         return __1__;
15     }
16 }q[N];
17
18 int main(){
19     cin>>n>>m;
20     for(int i=1;i<=n;i++){
21         cin>>a[i];
22         q[i].id=i;
23     }
24     for(int i=1;i<=m;i++){
25         int x,y;
26         cin>>x>>y;
27         __2__;
28     }
29     sort(q+1,q+n+1);
30     for(int i=1;i<=n;i++){
31         if(vis[q[i].id])continue;
32         Q.push(q[i].id);
33         int mx=0;
34         __3__;
35         while(!Q.empty()){
36             int x=Q.front();
37             Q.pop();
38             if(x!=q[i].id)mx=max(mx,a[x]);
```

```

39         for(int j=0;j<edge[x].size();j++){
40             int y=edge[x][j];
41             if(vis[y])continue;
42             ____4____;
43             vis[y]=1;
44             ____5____;
45         }
46     }
47 }
48 for(int i=1;i<=n;i++)cout<<a[i]<<" ";
49 cout<<endl;
50 }

```

35. 1 处应填

- | | |
|--------------------|---------------------|
| A. A.id==B.id | B. A.id<B.id |
| C. a[A.id]>a[B.id] | D. a[A.id]<a[B.id] |
| E. A.id>B.id | F. a[A.id]==a[B.id] |

36. 2 处应填

- | | |
|-------------------------|-------------------------|
| A. edge[x].push_back(y) | B. edge[y].push_back(i) |
| C. edge[i].push_back(y) | D. edge[i].push_back(x) |
| E. edge[x].push_back(i) | F. edge[y].push_back(x) |

37. 3 处应填

- | | |
|-------------------|----------------------|
| A. vis[i]=0 | B. vis[i]=1 |
| C. vis[mx]=1 | D. vis[a[q[i].id]]=1 |
| E. vis[q[i].id]=1 | F. vis[mx]=0 |

38. 4 处应填

- | | |
|----------------------|--------------------|
| A. a[y]=mx | B. a[x]=a[y] |
| C. a[y]=max(a[y],mx) | D. a[y]=a[q[i].id] |
| E. a[x]=mx | F. a[y]=a[x] |

39. 5 处应填

- | | |
|-----------------|-----------------------|
| A. Q.push(mx) | B. Q.push(edge[x][j]) |
| C. Q.push(a[y]) | D. Q.push(q[i].id) |
| E. Q.push(a[x]) | F. Q.push(y) |

2. 输入正整数 n, m ($n \leq 2000, m \leq 8000$)，点用 1 到 n 的整数编号。再输入 m 条有向边 u, v, w ，其中 $1 \leq u, v \leq n, |w| \leq 10^4$ 。输入任意两个点之间的最短路。

```

01 #include <cstdio>
02 #include <queue>
03 #include <cstring>

```

```

04 #include <algorithm>
05 using std::push_heap;
06 using std::pop_heap;
07 const int N = 2003, M = 8003, INF = 0x3f3f3f3f;
08 struct Edge {
09     int to, nxt, w;
10 } e[M << 1];
11 struct Item {
12     int id, dis;
13     bool operator<(const Item& i) const { return dis > i.dis; }
14 } heap[M];
15 int n, m, tot, pheap, head[N], h[N], dis[N], maxe[N];
16 bool vis[N];
17 std::queue<int> q;
18 void add(int u, int v, int w) {
19     e[++tot] = {v, head[u], w};
20     head[u] = tot;
21 }
22 void spfa() {
23     memset(h, 0x3f, sizeof h);
24     vis[0] = 1;
25     h[0] = 0;
26     q.push(0);
27     while (!q.empty()) {
28         int u = q.front();
29         q.pop(); vis[u] = 0;
30         for (int i = head[u]; i; i = e[i].nxt) {
31             int v = e[i].to, w = e[i].w;
32             if (h[v] > h[u] + w) {
33                 h[v] = h[u] + w;
34                 ____1____
35             }
36         }
37     }
38 }
39 void dijkstra() {
40     for (int i = 1; i <= n; ++i)
41         for (int j = head[i]; j; j = e[j].nxt)
42             ____2____;
43     for (int i = 1; i <= n; ++i) {
44         memset(dis, 0x3f, sizeof dis);
45         memset(vis, 0, sizeof vis);
46         pheap = dis[i] = 0;
47         heap[pheap++] = {i, 0};

```

```

48     push_heap(heap, heap + pheap);
49     for (int j = 1; j <= n; ++j) {
50         while(pheap && ____3____)
51             pop_heap(heap, heap + (pheap--));
52         if (!pheap) break;
53         int k = heap[0].id;
54         vis[k] = 1;
55         pop_heap(heap, heap + (pheap--));
56         for (int l = head[k]; l; l = e[l].nxt) {
57             int v = e[l].to;
58             if (!vis[v] && dis[v] > dis[k] + e[l].w) {
59                 dis[v] = dis[k] + e[l].w;
60                 heap[pheap++] = {v, dis[v]};
61                 push_heap(heap, heap + pheap);
62             }
63         }
64     }
65     for (int j = 1; j <= n; ++j)
66         if (dis[j] == INF)
67             printf("inf ");
68         else
69             printf("%d ", ____4____);
70     printf("\n");
71 }
72 }
73 void init() {
74     memset(maxe, 0x80, sizeof maxe);
75     scanf("%d%d", &n, &m);
76     for (int i = 1, u, v, w; i <= m; ++i) {
77         scanf("%d%d%d", &u, &v, &w);
78         add(u, v, w);
79         maxe[u] = std::max(maxe[u], w);
80     }
81     ____5____
82 }
83 int main() {
84     init();
85     spfa();
86     dijkstra();
87     return 0;
88 }

```

40. 1 处应填

A. q.push(u);

- B. `if (vis[v]) q.push(v);`
- C. `if (vis[v]) q.push(u);`
- D. `if (!vis[v]) { q.push(u); vis[v] = 1; }`
- E. `q.push(v);`
- F. `if (!vis[v]) { q.push(u); vis[v] = 0; }`

41. 2 处应填

- A. `e[j].w += h[i] - h[e[j].to]`
- B. `e[j].w += h[e[j].to] - h[i]`
- C. `e[j].w = h[i] + h[e[j].to] + e[j].w`
- D. `e[j].w = h[i] + h[e[j].to] - e[j].w`
- E. `e[j].w += h[i] + h[e[j].to] + e[j].w`
- F. `e[j].w = h[e[j].to] - h[i]`

42. 3 处应填

- A. `vis[heap[1].id]`
- C. `vis[heap[pheap - 1].dis]`
- B. `vis[heap[pheap - 1].id]`
- D. `vis[heap->id]`
- E. `vis[heap[pheap + 1].dis]`
- F. `vis[heap[pheap + 1].id]`

43. 4 处应填

- A. `dis[i] + h[j] - h[i]`
- C. `dis[j] + h[i] + h[j]`
- B. `dis[j] - h[i] + h[j]`
- D. `dis[j] + h[i] - h[j]`
- E. `dis[j] - h[i] - h[j]`
- F. `dis[i] - h[j] + h[i]`

44. 5 处应填

- A. `for (int i = 1; i <= n; ++i) add(0, i, maxe[i]);`
- B. `for (int i = 1; i <= n; ++i) add(0, i, 0);`
- C. `for (int i = 1; i <= n; ++i) if(maxe[i] < 0) add(0, i, 0);`
- D. `for (int i = 1; i <= n; ++i) if(maxe[i] < 0) add(0, i, maxe[i]);`
- E. `for (int i = 1; i <= n; ++i) add(0, maxe[i], 0);`
- F. `for (int i = 1; i <= n; ++i) if(maxe[i] < 0) add(0, maxe[i], 0);`