



C++ 八级 (样题)

1 单选题 (每题 2 分, 共 30 分)

题号	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
答案	B	C	D	B	C	B	C	C	C	D	B	A	C	C	C

第 1 题 从A城到C城需要经过B城, 其中A到B可选高铁和飞机, B到C可以自驾或打的, 请问A到C有几种交通选择 ()。

- A. 2
- B. 4
- C. 8
- D. 不知道

第 2 题 下面程序输出的n的值是 ()。

```

1 #include <iostream>
2 #include <string>
3 #include <cmath>
4 using namespace std;
5
6 int main()
7 {
8     int a,b,c,d, n=0;
9
10    for(a=1;a<5;a++)
11        for(b=1;b<5;b++)
12            for(c=1;c<5;c++)
13                for(d=1;d<5;d++)
14                    if(a!=b && a!=c && a!=d && b!=c && b!=d && c!=d)
15                        if(a!=4){
16                            cout << a*1000+b*100+c*10+d << " ";
17                            n++;
18                        }
19    cout << endl << n << endl;
20    return 0;
21 }

```

- A. 4
- B. 12
- C. 18
- D. 24

第 3 题 对 $(a + b)^5$,想求出 a^3b^2 的系数可以使用 ()

- A. 杨氏三角

- B. 祖冲之角
- C. 勾股三角
- D. 杨辉三角

第4题 对于4个结点的简单有向图，最少（）条边可以形成一条覆盖所有顶点的环。

- A. 5
- B. 4
- C. 3
- D. 2

第5题 对正整数 a 和 n (n 为 2 的正整数次幂)，下面求 a^n 值的方法是（）

```
1 int fan(int a, int n){
2     if(n==0) return 1;
3     if(n==1) return a;
4     long long s = fan(a, n/2);
5
6     return s*s;
7 }
```

- A. 折半
- B. 二分
- C. 倍增
- D. 迭代

第6题 平面内，通过一点可以作（）条平行于给定直线的直线？

- A. 0
- B. 1
- C. 2
- D. 无限多

第7题 定义常量 `const double pi=3.14159`。如果一个等边三角形的边长为4，下列（）表达式可以求其面积。

- A. $16*\sin(\pi/3)$
- B. $16*\cos(\pi/3)$
- C. $8*\sin(\pi/3)$
- D. $8*\cos(\pi/3)$

第8题 下面程序使用BFS遍历一个有 n 个顶点、边权都为1的无向图 G ，下面说法正确的是（）。

```

4#include <vector>
5using namespace std;
6
7#define N 2023
8
9int n,m;
10vector<int> G[N];
11int q[N],hd,tl;
12int dis[N];
13
14void BFS(int st)
15{
16    hd=1,tl=0;
17    for(int i=1;i<=n;i++)dis[i]=-1;
18    q[++tl]=st,dis[st]=0;
19    while(hd<=tl){
20        int u=q[hd++];
21        for(int i=0;i<G[u].size();i++){
22            int v=G[u][i];
23            if(dis[v]!=-1)continue;
24            dis[v]=dis[u]+1;
25            q[++tl]=v;
26        }
27    }
28}

```

- A. tl记录遍历的结点数
- B. dis按照贪心法变化
- C. dis存储st到其他顶点的距离
- D. 算法复杂度是 $O(n^2)$

第9题 下面的冒泡排序中尝试提前结束比较过程，横线处应该填写的代码是（）。

```

1void BubbleSort(int R[ ], int n)
2{
3    int i,j,lastExchangeIndex;
4
5    i = n;
6    while (i >1){
7        lastExchangeIndex = 1;
8        for (j = 1; j < i; j++)
9            if (R[j+1] < R[j]) {
10                int t;
11                t=R[j], R[j] = R[j+1], R[j+1] = t;
12                lastExchangeIndex = j;
13            } //if
14
15        _____ ;
16    } // while
17} // BubbleSort

```

- A. $i = \text{lastExchangeIndex} + 1$
- B. $i = \text{lastExchangeIndex} - 1$
- C. $i = \text{lastExchangeIndex}$
- D. $i = \text{lastExchangeIndex} - j$

第10题 对数列3、4、7、12、19、28、39求和，除简单累加外，还可以用下面（）来直接计算。

- A. 等差数列求和
- B. 等比数列求和
- C. 斐波拉契数列
- D. 其他某种有规律序列

第11题 约定杨辉三角形第0行只有1个元素是1，第1行有2个元素都是1，第四行的所有元素之和是（）？

- A. 8
- B. 16
- C. 24
- D. 32

第12题 下列程序的输出是（）。

```
1 int main()
2 {
3     int sum = 0, x;
4
5     for(x=-10; x<=10; x++)
6         if((3*x+5 >= -11) && (3*x+5 <= 11))
7             sum +=x;
8
9     cout << sum;
10
11    cout << endl;
12    return 0;
13 }
```

- A. -12
- B. 0
- C. 55
- D. -55

第13题 对于具有n个元素的二叉排序树（又名二分查找树）进行前序遍历，其时间复杂度是（）？

- A. O(1)
- B. O(log n)
- C. O(n)
- D. O(n^2)

第14题 Dijkstra的算法在实现时一般可以选用（）来提高效率？

- A. 数组
- B. 链表
- C. 堆

D. 栈

第 15 题 有关下面代码的说法正确的是 ()。

```
1 #include <iostream>
2
3 class Node {
4 public:
5     int Value;
6     Node* Next;
7
8     Node(int Val, Node* Nxt = nullptr) {
9         Value = Val;
10        Next = Nxt;
11    }
12 };
13
14 int main() {
15     Node* firstNode = new Node(10);
16     firstNode->Next = new Node(100);
17     firstNode->Next->Next = new Node(111, firstNode);
18     return 0;
19 }
```

- A. 上述代码构成单向链表。
- B. 上述代码构成双向链表。
- C. 上述代码构成循环链表。
- D. 上述代码构成指针链表。

2 判断题 (每题 2 分, 共 20 分)

题号	1	2	3	4	5	6	7	8	9	10
答案	√	×	√	√	√	×	√	√	√	×

第 1 题 学校组织班际排球赛, 每个班级可以派男女各一个参赛队伍, 每队 5 人。班级 A 的每位同学都可以报名, 那可以用加法原理来计算 A 班有多少支候选队伍。()

第 2 题 若 a, x, b 都是 double 类型, 则对方程 $a*x+b=0$ 求解的程序中可以直接用 $x=-b/a$ 来求解。()

第 3 题 从 15 本不同的书中选 3 本, 总共有 455 种方法。()

第 4 题 连通图 G 有 n 个顶点 m 条边, 须删除 $m-n+1$ 条边后才能变成一棵生成树。()

第 5 题 在 C++ 语言中, 所有 int 类型的值, 经过若干次左移操作 (<<) 后, 它们的值总会变为 0。()

第 6 题 如果一个四边形的对角线互相平分, 并且两条对角线的长度都为 8, 那么这个四边形的面积一定是 32。()

第 7 题 最小生成树的权值是指生成树所有边的权值之和最小。()

第 8 题 如果一个图中所有边的权重都为正数, 则 Floyd 算法可以求出该图中任意两个顶点间的最短路径。()

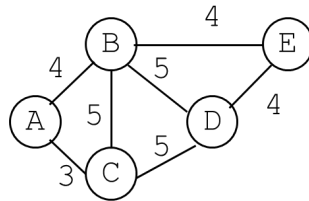
第 9 题 下面是图的深度遍历的代码, 则横线处可以填入: `if(vis[x]) return`。()

```

1 void dfs(int x) {
2     _____; // 补充代码
3     vis[x]=1;
4     cout<<x<<endl; // 访问当前节点
5     for(int i=vFirst[x]; i!=-1; i=e[i].next){// 遍历每一个相邻的顶点
6         dfs(e[i].v);
7     }
8 }

```

第 10 题 下图中A到E的Dijkstra单源最短路可以在第2次探索中找到。（ ）



3 编程题（每题 25 分，共 50 分）

3.1 编程题 1

- 试题名称：区间
- 时间限制：1.0 s
- 内存限制：128.0 MB

3.1.1 问题描述

小杨有一个长度为 n 的正整数序列 A 。

小杨有 q 次询问。第 i 次 ($1 \leq i \leq q$) 询问时，小杨会给出 l_i, r_i, x_i ，请你求出 x_i 在 $A_{l_i}, A_{l_i+1}, \dots, A_{r_i}$ 中出现的次数。

3.1.2 输入描述

第一行包含一个正整数 T ，表示数据组数。

对于每组数据：第一行包含一个正整数 n ，表示序列 A 的长度。第二行包含 n 个正整数 A_1, A_2, \dots, A_n ，表示序列 A 。第三行包含一个正整数 q ，表示询问次数。接下来 q 行，每行三个正整数 l_i, r_i, x_i ，表示一组询问。

3.1.3 输出描述

对于每组数据，输出 q 行。第 i 行 ($1 \leq i \leq q$) 输出一个非负整数，表示第 i 次询问的答案。

3.1.4 样例输入1

```
1 2
2 5
3 7 4 6 1 1
4 2
5 1 2 3
6 1 5 1
7 5
8 1 2 3 4 5
9 2
10 5 5 3
11 1 4 3
```

3.1.5 样例输出1

```
1 0
2 2
3 0
4 1
```

3.1.6 子任务

子任务编号	分值	n	q	$\max A_i$
1	30	≤ 100	≤ 100	≤ 10
2	30	$\leq 10^5$	$\leq 10^5$	$\leq 10^5$
3	40	$\leq 10^5$	$\leq 10^5$	$\leq 10^9$

对于全部数据，保证有 $1 \leq T \leq 5$, $1 \leq n \leq 10^5$, $1 \leq q \leq 10^5$, $1 \leq A_i \leq 10^9$ 。

3.1.7 参考程序

```
1 #include <bits/stdc++.h>
2 const int maxn=100086;
3 using namespace std;
4 int t, n, q;
5 map<int,vector<int> >mp;
6 int main()
7 {
8     ios::sync_with_stdio(false);
9     cin.tie(0);
10    cout.tie(0);
11    cin>>t;
12    while(t--){
13        cin>>n;
14        mp.clear();
15        for(int i = 1;i <= n;i++){
16            {
17                int x;
18                cin>>x;
19                mp[x].push_back(i);
20            }
21        }
22        cin>>q;
23        while(q--)
```

```
24     int l, r, x;
25     cin>>l>>r>>x;
26     cout<<upper_bound(mp[x].begin(), mp[x].end(), r) -
lower_bound(mp[x].begin(), mp[x].end(), l)<<endl;
27     }
28     }
29
30 }
```

3.2 编程题 2

- 试题名称: 小杨的旅游
- 时间限制: 1.0 s
- 内存限制: 128.0 MB

3.2.1 问题描述

小杨准备前往 B 国旅游。

B 国有 n 座城市，这 n 座城市依次以 1 至 n 编号。城市之间由 $n - 1$ 条双向道路连接，任意两座城市之间均可达（即任意两座城市之间存在可达的路径）。

小杨可以通过双向道路在城市之间移动，通过一条双向道路需要 1 单位时间。

B 国城市中有 k 座城市设有传送门。设有传送门的城市的编号依次为 b_1, b_2, \dots, b_k 。小杨可以从任意一座设有传送门的的城市花费 0 单位时间前往另一座设有传送门的的城市。

注：如果两座设有传送门的的城市之间存在双向道路，那么小杨可以选择通过双向道路移动，也可以选择通过传送门传送。

小杨计划在 B 国旅游 q 次。第 i 次旅游 ($1 \leq i \leq q$)，小杨计划从编号为 u_i 的城市前往编号为 v_i 的城市，小杨希望你能求出所需要的最短时间。

3.2.2 输入描述

第一行包含三个正整数 n, k, q ，分别表示 B 国的城市数量，设有传送门的的城市数量，以及小杨计划在 B 国旅游的次数。

接下来 $n - 1$ 行，每行包含两个正整数 x_i, y_i ，表示一条双向道路连接的两座城市的编号。

第 $n + 1$ 行包含 k 个正整数 b_1, b_2, \dots, b_k ，表示设有传送门的城市的编号。

接下来 q 行，每行包含两个正整数 u_i, v_i ，表示小杨第 i 次旅游行程的起点城市编号与终点城市编号。

3.2.3 输出描述

输出共 q 行。第 i 行 ($1 \leq i \leq q$) 输出一个非负整数，表示小杨计划第 i 次旅游从编号为 u_i 的城市前往编号为 v_i 的城市所需要的最短时间。

3.2.4 样例输入1

```
1 7 2 1
2 5 7
3 3 6
4 2 3
5 1 5
6 5 4
7 1 2
8 7 4
9 3 7
```

3.2.5 样例输出1

```
1 4
```

3.2.6 样例输入2

```
1 5 0 3
2 2 3
3 5 1
4 5 2
5 1 4
6 4 5
7 1 4
8 4 3
```

3.2.7 样例输出2

```
1 2
2 1
3 4
```

3.2.8 子任务

子任务编号	分值	n	k	q
1	30	≤ 500	≤ 500	$= 1$
2	30	$\leq 2 \times 10^5$	$= 0$	$\leq 2 \times 10^5$
3	40	$\leq 2 \times 10^5$	$\leq 2 \times 10^5$	$\leq 2 \times 10^5$

对于全部数据，保证有 $1 \leq k \leq n \leq 2 \times 10^5$ ， $1 \leq q \leq 2 \times 10^5$ ， $1 \leq x_i, y_i \leq n$ ， $1 \leq u_i, v_i \leq n$ 。对于所有 $1 \leq i \leq n-1$ ，有 $x_i \neq y_i$ 。

3.2.9 参考程序

```
1 #include<bits/stdc++.h>
2 using namespace std;
3 #define int long long
4 #define Getchar() p1==p2 and (p2=(p1=Inf)+fread(Inf,1,1<<21,stdin),p1==p2)?EOF:*p1++
5 #define Putchar(c) p3==p4 and (fwrite(0uf,1,1<<21,stdout),p3=0uf),*p3++=c
6 char Inf[1<<21],0uf[1<<21],*p1,*p2,*p3=0uf,*p4=0uf+(1<<21);
7 inline void read(int &x,char c=Getchar())
8 {
9     bool f=c!='-';
```

```

10     x=0;
11     while(c<48 or c>57) c=Getchar(),f&c!='-';
12     while(c>=48 and c<=57) x=(x<<3)+(x<<1)+(c^48),c=Getchar();
13     x=f?x:-x;
14 }
15 inline void write(int x)
16 {
17     if(x<0) Putchar('-'),x=-x;
18     if(x>=10) write(x/10),x%=10;
19     Putchar(x^48);
20 }
21 int
    n,k,q,head[200010],mini[200010],dep[200010],siz[200010],hvson[200010],fa[200010],start[200010],pos;
22 struct edge
23 {
24     int to,next;
25 };
26 edge e[400010];
27 inline void add(const int &x,const int &y)
28 {
29     static int cnt=0;
30     e[++cnt].to=y,e[cnt].next=head[x],head[x]=cnt;
31 }
32 inline void dfs1(int pos,int fath,int de,int maxi=-0x3f3f3f3f)
33 {
34     fa[pos]=fath,dep[pos]=de,siz[pos]=1;
35     for(int i=head[pos];i;i=e[i].next)
36         if(e[i].to!=fath)
37             {
38                 dfs1(e[i].to,pos,de+1),siz[pos]+=siz[e[i].to];
39                 if(siz[e[i].to]>maxi) hvson[pos]=e[i].to,maxi=siz[e[i].to];
40             }
41 }
42 inline void dfs2(int pos,int Start)
43 {
44     start[pos]=Start;
45     if(hvson[pos])
46         {
47             dfs2(hvson[pos],Start);
48             for(int i=head[pos];i;i=e[i].next) if(e[i].to!=fa[pos] &&
e[i].to!=hvson[pos]) dfs2(e[i].to,e[i].to);
49         }
50 }
51 inline int lca(int x,int y)
52 {
53     while(start[x]!=start[y])
54         if(dep[start[x]]>=dep[start[y]]) x=fa[start[x]];
55         else y=fa[start[y]];
56     if(dep[x]>=dep[y]) return y;
57     return x;
58 }
59 queue<int> qu;
60 inline void bfs()
61 {
62     while(!qu.empty())

```

```

63     {
64         pos=qu.front(),qu.pop();
65         for(int i=head[pos];i;i=e[i].next) if(mini[e[i].to]>mini[pos]+1)
mini[e[i].to]=mini[pos]+1,qu.push(e[i].to);
66     }
67 }
68 signed main()
69 {
70     read(n),read(k),read(q),memset(mini,0x3f,sizeof(mini));
71     for(int i=1,x,y;i<n;i++) read(x),read(y),add(x,y),add(y,x);
72     dfs1(1,0,1),dfs2(1,1);
73     for(int i=1,x;i<=k;i++) read(x),mini[x]=0,qu.push(x);
74     bfs();
75     for(int i=1,x,y;i<=q;i++)
76     {
77         read(x),read(y);
78         write(min(dep[x]+dep[y]-dep[lca(x,y)]*2,mini[x]+mini[y]),Puchar('\n'));
79     }
80     fwrite(Ouf,1,p3-Ouf,stdout),fflush(stdout);
81     return 0;
82 }

```