Shumen 2011.

1. (geometrija, trouglovi) Dato je N (3<=N<=26) tačaka u ravni nazivom i koordinatama. Kreirajte program **triangles** koji pronalazi sve trouglove čija tri temena mogu biti date tačke.

ULAZ	IZLAZ
Prva linija standardnog ulaza sadrži broj tačaka N. U sledećih N redova standardnog ulaza nalazi se naziv, apcisa i ordinata svake tačke. Apcisa i ordinata su celi brojevi iz intervala (–2000, 2000).	Ispisati u svakoj liniji standardnog izlaza nazive temena koji obrazuju trouglove. Temena svakog trougla ispisati u leksikografskom poretku. Ako zadate tačke ne obrazuju niti jedan trougao ispisati tekst ***No triangles***

TEST PRIMERI

ULAZ	IZLAZ
5 A 0 0 B 5 0 C 0 4 D 10 0 E 0 8	A B C A B E A C D A D E B C D B C E B D E C D E
4 X 0 0 Y 3 3 T 7 7 W 4 4	No triangles.
5 A 0 0 B 5 0 C 0 4 D 0 4 E 0 4	A B C A B D A B E
20 A 1 -17 B 3 3 C -66 -66 D 625 625 E 0 0 F 1 1 G 55 -55 H 33 33 I 1234 1234 J -90 -90 K 888 11 L 10 -17 M 3 36 N -66 -664 O 6251 625 P 0 10 Q 1 1 R 55 -555 T 33 331 Y 1234 1234 X -90 -902 Z 888 1198	A B C A B D A B E A B F A B G A B H A B I A B J A B K A B L A B M A B N A B O A B P A B Q A B R A B T A B Y A C D A C E A C F A C G A C H A C I

	A C J A C K A C L A C M A C N A C O A C P A C Q A C R A C T A C Y A D E A D F A D G A D H A D I A D J A D K A D L A D M A D N A D O A D P A D Q A D R A D T A D Y A E F A E G A E H A E I A E J A E K A E L A E M A E N A E O A E P A E Q A E R A E T A E Y A F G A F H A F I A F J A F K A F L A F M A F N A F O A F P A F R A F T A F Y A G H A G I

	A G J A G K A G L A G M A G N A G O A G P A G Q A G R A G T A G Y A H I A H J A H K A H L A H M A H N A H O A H P A H Q A H R A H T A H Y A I J A I K A I L A I M A I N A I O A I P A I Q A I R A I T A J K A J L A J M A J N A J O A J P A J Q A J R A J T A J Y A K L A K M A K N A K O A K P A K Q A K R A K T A K Y A L M A L N A L O A L P A L Q

	A L R A L T A L Y A M N A M O A M P A M Q A M R A M T A M Y A N O A N P A N Q A N R A N T A N Y A O P A O Q A O R A O T A O Y A P Q A P R A P T A P Y A Q R A Q T A Q Y A R T A R Y A T Y B C G B C K B C L B C M B C N B C O B C P B C R B C T B D G B D K B D L B D M B D N B D O B D P B D R B D T B E G B E K B E L B E M B E N B E O B E P B E R

	B E T B F G B F K B F L B F M B F N B F O B F P B F R B F T B G H B G I B G J B G K B G L B G M B G N B G O B G P B G Q B G R B G T B G Y B H K B H L B H M B H N B H O B H P B H R B H T B I K B I L B I M B I N B I O B I P B I R B I T B J K B J L B J M B J N B J O B J P B J R B J T B K L B K M B K N B K O B K P B K Q B K R B K T B K Y B L M

	B L N B L O B L P B L Q B L R B L T B L Y B M N B M O B M P B M Q B M R B M T B M Y B N O B N P B N Q B N R B N T B N Y B O P B O Q B O R B O T B O Y B P Q B P R B P T B P Y B Q R B Q T B R T B R Y B T Y C D G C D K C D L C D M C D N C D O C D P C D R C D T C E G C E K C E L C E M C E N C E O C E P C E R C E T C F G C F K C F L C F M C F N

	C F O C F P C F R C F T C G H C G I C G J C G K C G L C G M C G N C G O C G P C G Q C G R C G T C G Y C H K C H L C H M C H N C H O C H P C H R C H T C I K C I L C I M C I N C I O C I P C I R C I T C J K C J L C J M C J N C J O C J P C J R C J T C K L C K M C K N C K O C K P C K Q C K R C K T C K Y C L M C L N C L O C L P C L Q C L R C L T

	C L Y C M N C M O C M P C M Q C M R C M T C M Y C N O C N P C N Q C N R C N T C N Y C O P C O Q C O R C O T C O Y C P Q C P R C P T C P Y C Q R C Q T C R T C R Y C T Y D E G D E K D E L D E M D E N D E O D E P D E R D E T D F G D F K D F L D F M D F N D F O D F P D F R D F T D G H D G I D G J D G K D G L D G M D G N D G O D G P D G Q D G R

	D G T D G Y D H K D H L D H M D H N D H O D H P D H R D H T D I K D I L D I M D I N D I O D I P D I R D I T D J K D J L D J M D J N D J O D J P D J R D J T D K L D K M D K N D K O D K P D K Q D K R D K T D K Y D L M D L N D L O D L P D L Q D L R D L T D L Y D M N D M O D M P D M Q D M R D M T D M Y D N O D N P D N Q D N R D N T D N Y D O P

	D O Q D O R D O T D O Y D P Q D P R D P T D P Y D Q R D Q T D R T D R Y D T Y E F G E F K E F L E F M E F N E F O E F P E F R E F T E G H E G I E G J E G K E G L E G M E G N E G O E G P E G Q E G R E G T E G Y E H K E H L E H M E H N E H O E H P E H R E H T E I K E I L E I M E I N E I O E I P E I R E I T E J K E J L E J M E J N E J O E J P

	E J R E J T E K L E K M E K N E K O E K P E K Q E K R E K T E K Y E L M E L N E L O E L P E L Q E L R E L T E L Y E M N E M O E M P E M Q E M R E M T E M Y E N O E N P E N Q E N R E N T E N Y E O P E O Q E O R E O T E O Y E P Q E P R E P T E P Y E Q R E Q T E R T E R Y E T Y F G H F G I F G J F G K F G L F G M F G N F G O F G P F G R F G T

	F G Y F H K F H L F H M F H N F H O F H P F H R F H T F I K F I L F I M F I N F I O F I P F I R F I T F J K F J L F J M F J N F J O F J P F J R F J T F K L F K M F K N F K O F K P F K R F K T F K Y F L M F L N F L O F L P F L R F L T F L Y F M N F M O F M P F M R F M T F M Y F N O F N P F N R F N T F N Y F O P F O R F O T F O Y F P R F P T

	F P Y F R T F R Y F T Y G H I G H J G H K G H L G H M G H N G H O G H P G H Q G H R G H T G H Y G I J G I K G I L G I M G I N G I O G I P G I Q G I R G I T G J K G J L G J M G J N G J O G J P G J Q G J R G J T G J Y G K L G K M G K N G K O G K P G K Q G K R G K T G K Y G L M G L N G L O G L P G L Q G L R G L T G L Y G M N G M O G M P G M Q

	G M R G M T G M Y G N O G N P G N Q G N R G N T G N Y G O P G O Q G O R G O T G O Y G P Q G P R G P T G P Y G Q R G Q T G Q Y G R T G R Y G T Y H I K H I L H I M H I N H I O H I P H I R H I T H J K H J L H J M H J N H J O H J P H J R H J T H K L H K M H K N H K O H K P H K Q H K R H K T H K Y H L M H L N H L O H L P H L Q H L R H L T H L Y

	H M N H M O H M P H M Q H M R H M T H M Y H N O H N P H N Q H N R H N T H N Y H O P H O Q H O R H O T H O Y H P Q H P R H P T H P Y H Q R H Q T H R T H R Y H T Y I J K I J L I J M I J N I J O I J P I J R I J T I K L I K M I K N I K O I K P I K Q I K R I K T I L M I L N I L O I L P I L Q I L R I L T I M N I M O I M P I M Q I M R I M T I N O

	I N P I N Q I N R I N T I O P I O Q I O R I O T I P Q I P R I P T I Q R I Q T I R T J K L J K M J K N J K O J K P J K Q J K R J K T J K Y J L M J L N J L O J L P J L Q J L R J L T J L Y J M N J M O J M P J M Q J M R J M T J M Y J N O J N P J N Q J N R J N T J N Y J O P J O Q J O R J O T J O Y J P Q J P R J P T J P Y J Q R J Q T J R T J R Y

	J T Y K L M K L N K L O K L P K L Q K L R K L T K L Y K M N K M O K M P K M Q K M R K M T K M Y K N O K N P K N Q K N R K N T K N Y K O P K O Q K O R K O T K O Y K P Q K P R K P T K P Y K Q R K Q T K Q Y K R T K R Y K T Y L M N L M O L M P L M Q L M R L M T L M Y L N O L N P L N Q L N R L N T L N Y L O P L O Q L O R L O T L O Y L P Q L P R

	L P T L P Y L Q R L Q T L Q Y L R T L R Y L T Y M N O M N P M N Q M N R M N T M N Y M O P M O Q M O R M O T M O Y M P Q M P R M P T M P Y M Q R M Q T M Q Y M R T M R Y M T Y N O P N O Q N O R N O T N O Y N P Q N P R N P T N P Y N Q R N Q T N Q Y N R T N R Y N T Y O P Q O P R O P T O P Y O Q R O Q T O Q Y O R T O R Y O T Y P Q R P Q T P Q Y

	P R T P R Y P T Y Q R T Q R Y Q T Y R T Y

IDEJA: proveriti uređene trojke tačaka (i,j,k) i svka trojka tačaka koja nije kolinearna obrazuje trougao
Vremenska složenost provere kolineranosti za tri tačke je O(1)

Resenje 1:

```
#include <iostream>
using namespace std;

bool colinear(int x0, int y0, int x1, int y1, int x2, int y2)
{ int a1=x1-x0, a2=y1-y0;
  int b1=x2-x0, b2=y2-y0;
  return  a1*b2==a2*b1;
}

int main()
{
        char name[26];
  int x[26], y[26];

        int n;
        cin >> n;
        for(int i=0; i<n; i++)
                cin >> name[i] >> x[i] >> y[i];

  bool found = false;
        for(int i=0;i<n;i++)
          for(int j=i+1; j<n; j++)
     for(int k=j+1; k<n; k++)
       if(!colinear(x[i],y[i],x[j],y[j],x[k],y[k]))
       { cout <<name[i]<<" "<< name[j]<<" "<< name[k]<<endl;
         found = true;
                                  }

   if(!found) cout << "No triangles." << endl;
   return 0;
}
```

Resenje 2:

```
#include <iostream>
using namespace std;
struct point{
   int x;
   int y;
   char i;
};
point tacke[32];
int n,br=0;
double orient(point a,point b,point c){
   return a.x*b.y+a.y*c.x+b.x*c.y-(c.x*b.y+c.y*a.x+b.x*a.y);
}
```

```
bool pr(int a,int b,int c){
   return orient(tacke[a],tacke[b],tacke[c])!=0;
}
int main(){
   cin>>n;
   for (int i=0;i<n;++i){
      cin>>tacke[i].i;
      cin>>tacke[i].x;
      cin>>tacke[i].y;
   }
   for (int i=0;i<n;++i){
      for (int j=i+1;j<n;++j){
         for (int k=j+1;k<n;++k){
            if (pr(i,j,k)){cout<<tacke[i].i<<" "<<tacke[j].i<<" "<<tacke[k].i<<"\n";++br;}
         }
      }
   }if (br==0){cout<<"No triangles.\n";}
   return 0;
}
```

---

2. (stringovi, substrings) Dat je string *s*, dužine *n*, koji sadrži 2 slova *a* i *n*-2 slova *b*. Napišite program **`substring`**, koji za dato *n* i pozicije slova *a*, nalazi broj podstringova stringa *s*, koji sadrže bar jedno slovo *a*.

Ulaz	**Izlaz**
Prva i jedina linija standardnog ulaza sadrži redom brojeve *n*, *p*, *q*, gde *p* i *q* su pozicije slova *a* u stringu. Pozicije slova u stringu su numerisane redom 1,2, …,to *n*. **Ograničenja**: $1 < n < 10^6$, $1 \le p < q \le n$	Na standardni izlaz ispisati ostatak deljenja broja podstringova sa brojem 123456789.

Test primer

Ulaz	**Izlaz**
7 2 5	21
7000 2000 5000	16005000
76000 30000 70000	15131743
111000 70000 90000	80213486
400000 199999 200000	400363
999999 111111 888888	520203

Rešenje:

Ako string sadrži prvo slovo *a* (nalazi se na poziciji *p*), onda je njegov početak na nekoj od pozcija 1, 2, ..., *p*, a kraj mu je na nekoj od pozicija *p*, *p*+1, …, *n*. Broj ovih podniski je *p**(*n*–*p*+1). Ovde se računaju podstringovi koji sadrže drugo slovo *a*, kao i oni koji ga ne sadrže.

Ostaje da se prebroje podnizovi koji sadrže samo drugo slovo *a*. Njihov početak je na nekoj od pozicija *p*+1, *p*+2, …, *q*, a kraj na nekoj od pozicija *q*, *q*+1, …, *n*. Njihov broj je (*q*–*p*)*(*n*–*q*+1).

```
#include <iostream>
using namespace std;

const long long M = 123456789;

int main()
{ long long n,p,q;
  cin >> n >> p >> q;
```

```
  cout << (p*(n-p+1) + (q-p)*(n-q+1))%M << endl;
  return 0;
}
```

Resenje 2;

```
#include <iostream>
using namespace std;
long long n,p,q;
int main(){
   cin>>n>>p>>q;
   cout<<(p*(n-p+1)+q*(n-q+1)-p*(n-q+1))%123456789<<endl;
   return 0;
}
```

---

3. (grafovi DFS, company) Ilija je počeo da radi za veliku softversku kompanije. Hijerhija kompanije je u obliku stabla, gde svaka osoba (osim šefa) ima tačno jednog direktnog menadžera. Kompanija je podeljena u timove, tako da svaki programer i svi njegovi direktni i indirektni podređeni (ako postoje) formiraju tim. To znači da tim može biti sačinjen od drugih timova. Na primer u kompaniji poput Microsoft-a postoji tim koji radi na softverskom paketu Office, koji se sastoji od podtimova koji rade na softverski alatima Word, Excel, itd. U Ilijinom slučaju Stanko je šef kompanije, a njemu direktno su potčinjeni Ilija i Petar. Ilija je menadžer Kristijanu, Petar je menadžer Goranu i Tomi. Prema gore navedenim pravilima možemo zaključiti da Stanko, Ilija, Petar, Kristijan, Goran i Toma obrazuju tim. Ilija i Kristijan takođe formiraju tim, i Petar, Goran i Toma formiraju drugi tim.

Kompanija se upravo premestila u novu, veom dugačku, ali, na žalost, usku kalncelariju u kojoj postoji prostor za samo jedan red kompjuterskih stolova. Šef je već zauzeo krajnje levo mesto, i želi da svakom svom radniku napravi raspored sedenja tako da:

a) Direktni menadžer svakog programera sedi levo od programera.
b) Svi članovi tima zauzimaju uzastopne stolove (npr. sede jedan do drugog).

Ako razmotrimo navedeni primer programera Ilije i kolega, jedan moguć raspoed sedenja je, redom: Stanko, Petar, Goran, Toma, Ilija, Kristijan.

Pomozite Iliji da ostavi dobar utisak na svog šefa pisanjem programa, koji pronalazi moguć raspored sedenja programera (za datu hijerarhiju kompanije u obliku stabla).

Ulaz

U prvoj liniji standardnog ulaza biće dat ceo broj N - broj programera u kompaniji. Neka su programeri predstavljeni brojevima od 1 do N, gde je sa 1 označen šef kompanije (koji nema direktnog menadžera). U sledećiht N – 1 linija dati su parovi celih brojeva W_1 W_2, koji označavaju da programer sa brojem W_1 je direktan menadžer programera W_2.

Izlaz

Na standardni izlaz štampati jednu liniju koja predstavlja raspored sedenja programera i koja sadrži N celih brojeva između 1 i N koji su razdvojeni blankom. Ako postoji više od jednog rasporeda sedenja, ispisati onaj koji je leksikografski najmanji. Raspored A je leksikografski manji od rasporeda B, ako prvi (najlevlji) broj po kom se oni razlikuju je manji i nalazi se u rasporedu A. Na primer, raspored {1, 3, 4, 6, 5, 2, 7} je manji od rasporeda {1, 3, 5, 2, 4, 7, 6}.

Ograničenja

$1 \leq N \leq 200{,}000$

U 70% test primera, N će biti manji ili jednak od 10,000.

U 85% test primera, ne više od 200 programera će imati istog direktnog menadžera.

Test primeri:

Ulaz	**Izlaz**
6 1 2 2 5 4 3 1 4 4 6	1 2 5 4 3 6

14 9 11 1 9 1 3 3 8 1 2 2 4 2 5 10 13 3 7 9 10 2 6 10 12 11 14	1 2 4 5 6 3 7 8 9 10 12 13 11 14

Obrazloženje: U prvom test primeru Stanko je označen brojem 1, Ilija je 2, Kristijan je 5, Petar je 4, Goran je 3, Toma je 6.

Ideja:
Razmotrite drvo koje predstavlja hijererhiju kompanije. Možemo uočiti da direktni menadžer svakog programera će biti levo od njega, kao i indirektni supervizori. Ovo nas dovodi do zaključka da je ovo DFS pretraga grafa (DFS=pretraga grafa u dubinu) u svom tradicionalnom obliku. Dalje, kako nam je potrebno štampanje redosleda u leksikografskom poretku, izvršićemo sortiranje čvorova grafa.
Pošto je u 30% test primera N>=10000, MORA se voditi računa o dubini rekurzije tj.o veličinu steka.
Ukupna vremenska složenost je:
O(n) za učitavanje ulaza + O(n log n) za sortiranje naslednika + O(n) za rekurziju = O(n log n)

Rešenje 1:

```
#include <cstdio>
#include <vector>
#include <algorithm>

#define MAX 1048576

int main(void)
{
	FILE* in = stdin; FILE* out = stdout;
//	in = fopen("Company.in", "rt"); out = fopen("Company.out", "wt");

	int numNodes;
	static std::vector <int> v[MAX];

	fscanf(in, "%d", &numNodes);
	for (int edge = 0; edge < numNodes - 1; edge++)
	{
		int from, to;
		fscanf(in, "%d %d", &from, &to);
		v[from].push_back(to);
	}
	for (int i = 1; i <= numNodes; i++)
		std::sort(v[i].rbegin(), v[i].rend());

	static int ans[MAX], ansSize = 0;
	static int index[MAX] = {0};
	static int stack[MAX], stackSize = 0;
```

```
	stack[stackSize++] = 1;
	while (stackSize)
	{
		int node = stack[stackSize - 1];
		if (index[node] < (int)v[node].size())
			stack[stackSize++] = v[node][index[node]++];
		else ans[ansSize++] = stack[--stackSize];
	}
	for (int i = ansSize - 1; i >= 0; i--)
		fprintf(out, "%d%c", ans[i], i == 0 ? '\n' : ' ');

	return 0;
}
```

Rešenje 2:

```
#include <iostream>
#include <vector>
#include <algorithm>
using namespace std;
vector<int> naslednici[200004];
bool d[200004];
int n,a,b,c,p[200004],shef;

void end(int tek){
   cout<<tek<<" ";
   for (int i=0;i<naslednici[tek].size();++i){
      end(naslednici[tek][i]);
   }
}
int main(){
   cin>>n;
   for (int i=0;i<n-1;++i){
      cin>>a>>b;
      d[b]=true;
      naslednici[a].push_back(b);
   }
   for (int i=1;i<=n;++i){
      for (int j=0;j<naslednici[i].size();++j){
         p[j]=naslednici[i][j];
      }sort(p,p+naslednici[i].size());
      for (int j=0;j<naslednici[i].size();++j){
         naslednici[i][j]=p[j];
      }
   }
    for (int i=1;i<=n;++i){
      if (!d[i]){shef=i;break;}
   }
   end(shef);cout<<endl;
   return 0;
}
```