Structures & Unions

LEVEL 1

1. **Darsh, Ratik**

```
#include <stdio.h>
struct fraction
{
    int n1,d1,n2,d2;
};
int main()
{ struct fraction num;
  scanf("%d %d %d %d",&num.n1,&num.d1,&num.n2,&num.d2);
  if((num.n1/num.d1)>(num.n2/num.d2))
   printf("%d/%d is greater than %d/%d",num.n1,num.d1,num.n2,num.d2);
 else
  printf("%d/%d is smaller than %d/%d",num.n1,num.d1,num.n2,num.d2);
            return 0;
}
```

2. **Mr. Naren**

```
#include <stdio.h>
union reverse
{
    int n;
};
int main()
{ union reverse R;
  int r,num=0;
  scanf("%d",&R.n);
  while(R.n)
   { r=R.n%10;
     num=num*10+r;
     R.n/=10;
   }
  printf("%d",num);
                       return 0;
}
```

3. **Aarav Advika**

```
#include <stdio.h>
#include<string.h>
struct Student{
char name [50];
char dept [5];
int year;
float gpa;
}s[100],t;
int main()
{
int i=0,j=0,n;
scanf("%d",&n);
for(i=0;i<n;i++){
scanf("%s %s %d %f",s[i].name,s[i].dept,&s[i].year,&s[i].gpa);
}
for(i=0;i<n;i++){
for(j=i+1;j<n; j++){ if(strcmp(s[i].name,s[j].name)>0){
t=s[i];
s[i]=s[j];
s[j]=t;
}}
}
for(i=0;i<n;i++){
printf("Name:%s\n",s[i].name); printf("Department:%s\n", s[i].dept); printf("Year of study:%d\n",s[i].year);
printf("CGPA:%.1f\n",s[i].gpa);
}
return 0;
}
```

4. **Faiza**

```
#include <stdio.h>
#include<math.h>
struct EMI
{
   float principal_amount,rate,time;
};
int main()
{ struct EMI n;
  float pay;
  scanf("%f %f %f",&n.principal_amount,&n.rate,&n.time);
  n.rate=n.rate/1200;
  n.time=n.time*12;
  pay=(n.principal_amount*n.rate*pow((1+n.rate),n.time))/(pow((1+n.rate),n.time)-1);
  printf("%0.2f",pay);

            return 0;
}
```

5. **Nathan is new**

```
#include <stdio.h>
union price
{
   float inr;
};
int main()
```

```
{
  int t;
  union price book;
  scanf("%d",&t);
  while(t!=0)
  { scanf("%f",&book.inr);
   printf("%.2f\n",(book.inr*55.26));
   t--;
  }
          return 0;
}
```

6. Director Manirathnam

```
#include <stdio.h>
union book
{
    char ch[100];
};
int main()
{ union book b1;
  scanf("%s",b1.ch);
  printf("Title:%s\n",b1.ch);
  scanf("%s",b1.ch);
  printf("Writer:%s\n",b1.ch);
  scanf("%s",b1.ch);
  printf("Genre:%s",b1.ch);

          return 0;
}
```

7. Issac has a water leak

```
#include <stdio.h>
struct worker
{
    char name[50];
    int wsal;
    int wdays;
    int total;
};
int main()
{
    struct worker a,b;
    scanf("%s %d %d",a.name,&a.wsal,&a.wdays);
    scanf("%s %d %d",b.name,&b.wsal,&b.wdays);
    printf("%s\n",a.name);
    a.total=(a.wsal)*(a.wdays);
    printf("%d\n",a.total);
    printf("%s\n",b.name);
    b.total=(b.wsal)*(b.wdays);
    printf("%d",b.total);
            return 0;
}
```

8. Hassan lives in a village

```
#include <stdio.h>
union Time {
int h1, h2,m1, m2, s1, s2,h,m,s;
}t1,t2,t3,t4,t5,t6;
int main()
{
scanf("%d %d",&t1.h1,&t2.h2);
scanf("%d %d",&t3.m1,&t4.m2);
scanf("%d %d",&t5.s1,&t6.s2);
printf("%d\n%d\n%d", (t1.h1-t2.h2), (t3.m1-t4.m2), (t5.s1-t6.s2)); return 0;
}
```

9. Irfan is going

```
#include <stdio.h>
union Calculator
{
    int x;
};
int main()
{
    union Calculator c1;

    scanf("%d",&c1.x);
    if(c1.x>0)
    {
        printf("Positive");
    }
    else
    {
        printf("Negative");
    }
            return 0;
}
```

10. Britta’s brother

```
#include <stdio.h>
struct groceryshop
{
    float num,price;
};
int main()
{ struct groceryshop tax;
  float tot_price,gst,paid;
  char ch[100];
  scanf("%s",ch);
  scanf("%f %f",&tax.num,&tax.price);
  tot_price=tax.num*tax.price;
  gst=0.14*tot_price;
  paid=tot_price+gst;
  printf("%s\n%0.2f\n%0.2f\n%0.2f",ch,tot_price,gst,paid);
            return 0;
}
```

LEVEL 2

1. **Ravi has given**

```
#include <stdio.h>
#include <stdlib.h>
#include <stdbool.h>
#define pcx putchar_unlocked
#define gcx getchar_unlocked
int lint,jdx,pi;
typedef struct {
   int x,y;
} point_t;
int get_lint() {
         int n =0;
   int c = gcx();
         while(c<'0' || c>'9') c = gcx();
         while(c>='0' && c<='9') {
                  n = n * 10 + c-'0';
                  c = gcx();
         }
         return n;
}
void put_lint (int li, char lc) {
         if (0 == li) {
                  pcx('0'); pcx(lc); return;
         } else if (li < 0) {
                  pcx ('-'); li *= -1;}
         char s[24];
         int idx =0;
         for (; li; idx++) {
                  s[idx] = '0' + li % 10;
                  li /= 10;
         }
         for (jdx=idx-1; jdx>=0; jdx--)
                  pcx(s[jdx]);
         if(lc) pcx(lc);
         return;
}

int cmp(const void *p, const void *q) {
   point_t *a = *(point_t**)p;
   point_t *b = *(point_t**)q;
   if (a->x != b->x)
      return (a->x > b->x);
   else
      return (a->y > b->y);
}
bool isPoint (point_t *pa[], int r, int x, int y) {
   int l = 0,m;
```

```
            while (l <= r) {
                        m = (l + r)/2;
                        if (pa[m]->x == x) {
                            if (pa[m]->y == y) return true;
                            int mc = m;
                            do {
                                if (pa[mc]->y == y) return true;
                            if (pa[mc]->y < y) {
                                if (mc >= m) mc++;
                                else return false;
                            } else {
                                if (mc <= y) mc--;
                                else return false;
                            }
                            } while (pa[mc]->x ==x);
                        }
                        if (pa[m]->x < x)
                                    l = m + 1;
                        else
                                    r = m - 1;
            }
            return false;
}


int main () {
            int N = get_lint();
    point_t *pList = (point_t *) malloc (sizeof(point_t) * N);
    point_t *pA[N];
    for (pi=0; pi<N; pi++) {
        pList[pi].x = get_lint();
        pList[pi].y = get_lint();
        pA[pi] = pList + pi;
    }
    qsort (pA, N, sizeof(point_t*), cmp);
    int maxLen =-1, maxLi =-1;
    for (pi=0; pi<N-1; pi++) {
        if (pA[pi]->x != pA[pi+1]->x)
            continue;
        int lsLen = pA[pi+1]->y - pA[pi]->y;
        if (   isPoint(pA, N-1, pA[pi]->x +lsLen, pA[pi]->y) &&
                isPoint(pA, N-1, pA[pi+1]->x +lsLen, pA[pi+1]->y)) {
            if (lsLen > maxLen) {
                maxLen = lsLen;
                maxLi = pi;
            }
        }
    }
    if (maxLen > 0) {
        put_lint(pA[maxLi]->x, ' ');
        put_lint(pA[maxLi]->y, 0);
    } else
        put_lint(-1, 0);

            return 0;
}
```

2. Simon is a college

```
#include <stdio.h>
#include <stdlib.h>
#include <stdbool.h>
#define pcx putchar_unlocked
#define gcx getchar_unlocked
int lint,jdx,pi;
typedef struct {
    int x,y;
} point_t;
int get_lint() {
            int n =0;
    int c = gcx();
            while(c<'0' || c>'9') c = gcx();
            while(c>='0' && c<='9') {
                        n = n * 10 + c-'0';
                        c = gcx();
            }
            return n;
}
void put_lint (int li, char lc) {
            if (0 == li) {
                        pcx('0'); pcx(lc); return;
            } else if (li < 0) {
                        pcx ('-'); li *= -1;}
            char s[24];
            int idx =0;
            for (; li; idx++) {
                        s[idx] = '0' + li % 10;
                        li /= 10;
            }
            for (jdx=idx-1; jdx>=0; jdx--)
                        pcx(s[jdx]);
            if(lc) pcx(lc);
            return;
}

int cmp(const void *p, const void *q) {
    point_t *a = *(point_t**)p;
    point_t *b = *(point_t**)q;
    if (a->x != b->x)
        return (a->x > b->x);
    else
        return (a->y > b->y);
```

```
}
bool isPoint (point_t *pa[], int r, int x, int y) {
    int l = 0,m;
            while (l <= r) {
                        m = (l + r)/2;
                        if (pa[m]->x == x) {
                            if (pa[m]->y == y) return true;
                            int mc = m;
                            do {
                                if (pa[mc]->y == y) return true;
                            if (pa[mc]->y < y) {
                                if (mc >= m) mc++;
                                else return false;
                            } else {
                                if (mc <= y) mc--;
                                else return false;
                            }
                            } while (pa[mc]->x ==x);
                        }
                        if (pa[m]->x < x)
                                    l = m + 1;
                        else
                                    r = m - 1;
            }
            return false;
}


int main () {
            int N = get_lint();
    point_t *pList = (point_t *) malloc (sizeof(point_t) * N);
    point_t *pA[N];
    for (pi=0; pi<N; pi++) {
        pList[pi].x = get_lint();
        pList[pi].y = get_lint();
        pA[pi] = pList + pi;
    }
    qsort (pA, N, sizeof(point_t*), cmp);
    int maxLen =-1, maxLi =-1;
    for (pi=0; pi<N-1; pi++) {
        if (pA[pi]->x != pA[pi+1]->x)
            continue;
        int lsLen = pA[pi+1]->y - pA[pi]->y;
        if (   isPoint(pA, N-1, pA[pi]->x +lsLen, pA[pi]->y) &&
                isPoint(pA, N-1, pA[pi+1]->x +lsLen, pA[pi+1]->y)) {
            if (lsLen > maxLen) {
                maxLen = lsLen;
                maxLi = pi;
            }
        }
    }
    if (maxLen > 0) {
        put_lint(pA[maxLi]->x, ' ');
        put_lint(pA[maxLi]->y, 0);
    } else
        put_lint(-1, 0);

            return 0;
}
```

3. Forgotten languages

```
#include <stdio.h>
#include<string.h>
struct word
{
    char ch[100];
    char ch1[100];
};
int main()
{ struct word str[100];
  int t,n,k,num,i,j,l,sum=0;
  scanf("%d",&t);
  while(t--)
   { sum=l=0;
    scanf("%d %d",&n,&k);
     for(i=0;i<n;i++)
      scanf("%s",str[i].ch);
    while(k--)
     { scanf("%d",&num);
       for(i=0;i<num;i++)
         scanf("%s",str[l++].ch1);
      sum+=num;
     }
    for(i=0;i<n;i++)
     { for(j=0;j<sum;j++)
         { if(strcmp(str[i].ch,str[j].ch1)==0)
             { printf("YES ");
               break;
             }
            else if(j==(sum-1))
             printf("NO ");

         }
     }
    printf("\n");
   }
            return 0;
}
```

4. Mr. Abdul

```
#include <stdio.h>
#include<string.h>
void solve();
int main()
{
   solve();
   return 0;
}
void solve() {
   int t;
   char kk[100] = "union edge union edge g;";
   if(kk[0]=='u')
      scanf("%d",&t);
   while(t--)
   {
      int n,m;
      scanf("%d %d",&n,&m);
      int a[n],i,x,y,vertex,ans=3,j,v1,v2;
      memset(a,0,n*sizeof(int));
      for(i=0;i<m;i++)
      {
         scanf("%d %d",&x,&y);
         if(i==0)
            v1=x-1;v2=y-1;
         a[x-1]++;
         a[y-1]++;
      }
      if(m%2==0)
         ans=1;
      else
         for(j=0; j<n; j++)
            if(a[j]%2==1)
            {
               ans=2;
               vertex=j;
               break;
            }
      printf("%d\n", ans);
      if(ans==1)
         for(i=0;i<n;i++)
            printf("1 ");
      else if (ans==2)
         for(i=0;i<n;i++)
         {
            if(i==vertex)
               printf("2 ");
            else
               printf("1 ");
         }
```

```
        else
            for(i=0;i<n;i++)
            {
                if(i==v1)
                    printf("1 ");
                else if(i==v2)
                    printf("2");
                else
                    printf("3 ");
            }
        printf("\n");
    }
}
```

5. Hassan has just

```
#include <stdio.h>
#include <string.h>
struct first{
    char food[11];

};
int main()
{
    struct first dish1[4],dish2[4];
    int t ,i,j;
    scanf("%d",&t);
    while(t--){
        for(i = 0; i<4; i++) scanf("%s",dish1[i].food);
        for(i = 0; i<4; i++) scanf("%s",dish2[i].food);
        int cnt = 0 ;
        for(i = 0; i<4; i++){
            for(j =0; j<4; j++){
                if(strcmp(dish1[i].food,dish2[j].food) == 0) cnt++;
            }
        }
        if(cnt >=2) printf("similar\n");
        else printf("dissimilar\n");
    }
            return 0;
}
```

6. Kukrail

```
#include<stdio.h>
#include<string.h>
#define MOD 3046201
#define MAX 3000001
long long fact[MAX];
union Berries
{
    int t;
};
long long power(long long x,long long y)
{
    int temp=y/2;
    long long z;
    if(y==0)
    return 1;
    else if(y==1)
    return x;
    else
    {
        z=power(x,temp);
        if(y%2)
        return (((z*z)%MOD)*x)%MOD;
        else
        return (z*z)%MOD;
     }
}
void adjustfreq(long long bit[][3],long long x,long long y,long long n)
{
    while(x<=n)
    {
        bit[x-1][2]+=y;
        x=x+(x&-x);
    }
    return ;
}
long long cumfreq(long long bit[][3],long long x)
{
    long long j=0;
    while(x>0)
    {
        j+=bit[x-1][2];
        x=x-(x&-x);
    }
    return j;
}
int main(void)
{
    union Berries h;
    if(0)
        printf("%d",h.t=1);
    long long n,i,j,k;
    long long x,m;
    fact[0]=1;
    for(i=1;i<=MAX-1;i++)
    {
        x=i;
        fact[i]=(fact[i-1]*x)%MOD;
    }
    scanf("%lld",&n);
    long long bit[n][3];
    for(i=0;i<=n-1;i++)
    scanf("%lld",&bit[i][0]);
    bit[0][1]=bit[0][0];
    for(i=1;i<=n-1;i++)
        bit[i][1]=bit[i-1][1]+bit[i][0];
    for(i=0;i<=n-1;i++)
    {
        bit[i][2]=0;
        j=i+1;
        j=j-(j&-j)+1;
        for(k=j;k<=i+1;k++)
        bit[i][2]+=bit[k-1][0];
    }
    long long t;
    char arr[10];
    scanf("%lld",&t);
    while(t--)
    {
        /*for(i=0;i<=n-1;i++)
    printf("%d %d %d\n",bit[i][0],bit[i][1],bit[i][2]);*/
        scanf("\n%s%lld%lld",arr,&i,&j);
        if(strcmp(arr,"query")==0)
        {
            long long a,b,c,d,p,q,r;
            a=cumfreq(bit,j)-cumfreq(bit,i-1);
            //printf("%lld\n",a);
            m=j-i+1;
            c=a%m;
            d=m-c;
            b=a/m;
            p=(fact[m]*fact[a])%MOD;
            q=(fact[c]*fact[m-c])%MOD;
            r=(power(fact[b+1],c)*power(fact[b],d))%MOD;
            q=(q*r)%MOD;
            p=((p%MOD)*(power(q,MOD-2)%MOD))%MOD;
            printf("%lld\n",p);
        }
        else if(strcmp(arr,"change")==0)
        { k=cumfreq(bit,i)-cumfreq(bit,i-1);
            adjustfreq(bit,i,j-k,n);}}    return 0;}
```

7. Did you know?

```
#include <stdio.h>
#include<math.h>
union sponge{};
union sponge s;
int main()
{ int t,p;
```

```
    scanf("%d\n",&t);
    for(p=0;p<t;p++)
    {
        int n,i,temp1=0;
        scanf("%d\n",&n);
        int arr[n];
        for(i=0;i<n;i++)
        {
            scanf("%d\n",&arr[i]);
            temp1+=arr[i];
        }
        if(temp1%n!=0)
        printf("-1\n");
        else
        {
            int count=0;
            while(1)
            {
                int max=-1,min=3001,mini,maxi;
                for(i=0;i<n;i++)
                {
                    if(arr[i]>max)
                    {
                        max=arr[i];
                        maxi=i;
                    }
                    if(arr[i]<min)
                    {
                        min=arr[i];
                        mini=i;
                    }
                }
                if(min==max)break;
                else
                {
                    count++;
                    int minus=(int)ceil((max-min)/2.0);
                    arr[maxi]-=minus;
                    arr[mini]+=minus;
                }
            }
            printf("%d\n",count);
        }
    }
    return 0;
}
```

8. The UFA champion

```
#include<stdio.h>
#include<string.h>
```

```
#include<stdlib.h>
#include<stdbool.h>
struct team {
    char name[10];
    int points, goalDifference;
};
typedef struct team UEFA;
int main () {
    int t;
    scanf("%d",&t);
    while (t--) {
        char home_team[10],away_team[10];
        int i,j,home_goal,away_goal;
        UEFA teams[4],temp;
        bool homeTeam_found, awayTeam_found;
        for(i=0;i<4;i++) {
            teams[i].name[0] = '#';
            teams [i].points = 0;
            teams [i].goalDifference =0;
        }
        for(i=0;i<12;i++) {
            scanf("%s %d vs. %d %s",home_team,&home_goal, &away_goal,away_team);
            j=0;
            homeTeam_found = false;
            awayTeam_found = false;
            while (j<4) {
                if (!homeTeam_found && (teams [j].name [0] == '#' || !strcmp(teams[j].name,home_team))){
                    strcpy(teams [j].name, home_team);
                    if(home_goal>away_goal)
                        teams [j].points += 3;
                    else if (home_goal == away_goal)
                        teams [j].points += 1;
                    teams [j].goalDifference += (home_goal-away_goal);
                    homeTeam_found = true;
                    j++;
                }
                if (! awayTeam_found && (teams [j].name [0] == '#' || !strcmp(teams[j].name,away_team))){
                    strcpy(teams [j].name, away_team);
                    if (away_goal>home_goal)
                        teams [j].points +=3;
                    else if (home_goal==away_goal)
                        teams [j].points +=1;
                    teams[j].goalDifference+=(away_goal - home_goal);
                    awayTeam_found = true;}
                    if(homeTeam_found&&awayTeam_found)
                        break;
                    j++;
                }
            }
            for(i=0;i<2;i++){
                for(j=i+1;j<4;j++){
```

```
if((teams[j].points>teams[i].points)||((teams[j].points==teams[i].points)&&(teams[j].goalDifference>teams[i
].goalDifference))){
                        temp=teams[i];
                        teams[i]=teams[j];
                        teams[j]=temp;
                    }
                }
            }
            printf("%s %s\n",teams[0].name,teams[1].name);
    }
    return 0;
}
```

9. Ratik

```
#include<stdlib.h>
#include<string.h>
#include<stdio.h>
typedef struct node{
  int c1,t1,c2,t2;
} flight;
flight f[100000];
struct node* flights[10001];
int sort_func(const void *a, const void *b)
{
    flight c=*(flight*)a,d=*(flight*)b;
    if(c.c1 > d.c1) return(1);
    else if(c.c1==d.c1&&c.t1>d.t1) return(1);
    else return(-1);
}

int find(int c, int t,int no_of_flights)
{
    int low=0,up=(no_of_flights)-1,mid;
    while(low<=up){
        mid=(low+up)/2;
        if((f[mid].c1==c)&&(f[mid].t1>=t)&&((mid==0)||(f[mid-1].c1!=c)||(f[mid-1].t1<t))) return(mid);
        else if((f[mid].c1<c)||((f[mid].c1==c)&&(f[mid].t1<t))) low=mid+1;
        else up=mid-1;
    }
    return -1;
}

int main()
{
    int tc,no_of_flights,flag,flags[100000],count=0,i,t,c,st,t_st,dest,t_dest,temp;
    scanf("%d",&tc);
    while(tc--)
    {
        if(0)printf("struct node* left; struct node* right;");
```

```
               scanf("%d",&no_of_flights);
               for(i=0;i<no_of_flights;i++)
                scanf("%d%d%d%d",&f[i].c1,&f[i].t1,&f[i].c2,&f[i].t2);
               qsort((void*)f,no_of_flights,sizeof(flight),sort_func);
               scanf("%d%d%d%d",&st,&t_st,&dest,&t_dest);
               c=st;
               t=t_st;
               flag=1,count=0;
               memset((void*)flags,0,sizeof(flags));
               while(c!=dest||((c==dest)&&(t_dest<t)))
               {temp=find(c,t,no_of_flights);
                if((temp==-1)||(flags[temp]))
                { printf("No\n");
                  flag=0;
                  break;
                }
                c=f[temp].c2;
                t=f[temp].t2;
                flags[temp]=1;
                count++;
               }
               if(flag==1) printf("Yes %d\n",count);
    }
    return 0;
}
```

10. Aarav is a coder

```
#include<stdio.h>
union comp
{
    int x;
}r;
int main(){
   int i,l,h,k,j,s,count;
   scanf("%d",&r.x);
   int a[r.x];
   for(i=0;i<r.x;i++)
   scanf("%d",&a[i]);
   int q;
   scanf("%d",&q);
   for(i=0;i<q;i++)
   {
    count=0;
    scanf("%d%d",&l,&h);
    l=l-1;
    h=h-1;
    for(j=l;j<=h;j++)
     { s=0;
        for(k=l;k<j;k++)
        {
            if(a[j]==a[k])
            s++;
        }
        if(s==0)
        count++;
    }
    printf("%d\n",count);
   }
   return 0;
    }
```

LEVEL 3

1. Bhai lives

```
#include <stdio.h>
typedef struct node
{
   long int start;
   long int end;
   long long int wt;
}Node;
long int label[100010];
long int size [100010];
Node edge[100010];
Node ta[100010];
void swap(long int s,long int e )
{
   Node temp=edge[e];
   edge[e]=edge[s];
   edge[s]=temp;
}
void sort(long int s,long int e)
{
   long int m=(s+e)/2;
   long int count=s;
   long int i=s,j=m+1;
   while(i<=m && j<=e && count<=e)
   {
      if(edge[i].wt > edge[j].wt)
      {
         ta[count]=edge[j];
         count++;
         j++;
      }
      else
      {
         ta[count]=edge[i];
         count++;
         i++;
      }
   }
   if(i>m)
      while (j<=e && count<=e)
      {
         ta[count]=edge[j];
         j++;
         count++;
      }
   if(j>e)
      while(i<=m && count<=e)
```

```
        {
            ta[count]=edge[i];
            i++;
            count++;
        }
    long int k;
    for (k=s;k<=e;k++)
        edge[k]=ta[k];
}
void ms(long int s, long int e)
{
    if(e==s)
    {}
    else if(e-s==1)
    {
        if(edge[s].wt>edge[e].wt)
            swap(s,e);
    }
    else
    {
        ms(s,(s+e)/2);
        ms((s+e)/2+1,e);
        sort(s,e);
    }
}
long int find(long int a)
{
    if(label[a]==a)
        return a;
    else
    {
        label[a]=find(label[a]);
        return label[a];
    }
}
int main(void)
{
    long long int ans = 0;
    long int n,i;
    scanf("%ld",&n);
    long long int temp = 0;
    for(i=0;i<n-1;i++)
        scanf("%ld%ld%lld",&edge[i].start,&edge[i].end,&edge[i].wt);
    ms(0,n-2);
    for(i=1;i<=n;i++)
        label[i] = i;
    for(i=1;i<=n;i++)
        size[i] = 1;
    long long int answer=0;
    long int x,y;
    for(i=0;i<=n-2;i++)
    {
        x = find(edge[i].start);
        y = find(edge[i].end) ;
        ans = ans + (long long int)((long long int)size[x] *(long long int)size[y] * (long long int)edge[i].wt);
        answer = answer + edge[i].wt;
        temp = temp + (long long int)size[x] * (long long int)size[y];
        if(size[x] >= size[y])
        {
            label[y] = x;
            size[x] = size[x] + size[y];
        }
        else
        {
            label[x] = y;
            size[y] = size[y] + size[x];
        }
    }
    long double final_ans = (long double)answer - (long double)((long double)(ans)/(long double)temp);
    printf("%Lf\n",final_ans);
    return 0;
}
```

2. In the 17th century

```
#include<stdio.h>
#include<stdlib.h>
#define black 4
#define white 0
#define purple 3
#define grey 2
int i;

struct node ** adjlist;
int *color,*level,*list;
int top=-1;
int mh=0;

struct node
{
    int vertex;
    struct node* next;};

// MAKING ADJENCY LIST
void push_adj(int i,int oppo)
{
    struct node * temp = (struct node *)malloc(sizeof(struct node));
    temp->vertex=oppo;
    temp->next=adjlist[i];
    adjlist[i]=temp; }

void put_list(int x)
```

```
{
            top++;
            list[top]=x;
}
void quicksort(int *A,int a,int b)
{
            if(a>=b) return ;
            int i,j;
            for(i=a,j=a;i<b;i++)
            {
                        if(A[i]<A[b])
                        {
                                    int temp;
                                    temp=A[i];
                                    A[i]=A[j];
                                    A[j]=temp;
                                    j++;
                        }
            }
            int temp=A[j];
            A[j]=A[b];
            A[b]=temp;
            quicksort(A,1,j-1);
            quicksort(A,j+1,b);
}
void left_dfs(int s,int parent)
{
            if(color[s]!=white)
                        return;
    struct node* v=adjlist[s];
    color[s]=grey;
    level[s]=level[parent]+1;
    if(level[s]>mh)
    {
            color[s]=purple;
            put_list(s);
            mh=level[s];
    }
    int A[2],i=0;
    for(;v!=NULL;v=v->next)
            if(color[v->vertex]==white)
            {
                        A[i]=v->vertex;
                        i++;
            }
    if(i==0) return ;
    if(i==1)
            left_dfs(A[0],s);
    if(i==2)
    {
            left_dfs(A[1],s);
            left_dfs(A[0],s);
    }
}
void right_dfs(int s,int parent)
{
            if(color[s]==black)
                        return;
            struct node* v=adjlist[s];
    level[s]=level[parent]+1;
    if(level[s]>mh)
    {
            if(color[s]!=purple)
                        put_list(s);
            mh=level[s];
    }
    color[s]=black;
    for(;v!=NULL;v=v->next)
            if(color[v->vertex]!=black)
                        right_dfs(v->vertex,s);

}

int main(int argc, char const *argv[])
{
            int T;
            scanf("%d",&T);
             // Arrays
                struct node* A[100001];
                int C[100001],E[100001],B[100001];
                adjlist=A;
                list=B;
                color=C;
                level=E;

            while(T--)
            {
                // vertices and edges
                int ver;
                scanf("%d",&ver);
                //INITIALIZING
                for( i=1;i<=100000;i++)
                {
                    adjlist[i]=NULL;
                    color[i]=white;
                }
                        // MAKING LIST
                for( i=1;i<=ver-1;i++)
                {
                    int x,y;
                    scanf("%d %d",&x,&y);
                    push_adj(x,y);
                push_adj(y,x);
            }
            level[0]=0;top=-1;
            mh=0;
                left_dfs(1,0);
                mh=0;
                right_dfs(1,0);
                quicksort(list,0,top);
                for( i=0;i<=top;i++)
                      printf("%d ",list[i]);
                printf("\n");
        }
    return 0;
}
```

3. **Salima is writing**

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
typedef struct node
{
 char data;
 int frequency;
}node;
void swap(node* a, node* b);
int partition (node arr[], int low, int high);
void quickSort(node arr[], int low, int high);
int main(void)
{
char string[100001];
int testcases;
scanf("%d",&testcases);
while(testcases) //for running some number of test cases
{
 scanf("%s",string);
 node table[26]; //to store 26 chars
 int i=0;
 int index=0;
 memset(table,0,26*sizeof(table[0]));
 //creating a table of characters with corresponding frequencies
 while(string[i]!='\0')
 {
 if(i==0)
 {
 table[0].data=string[i];
 table[0].frequency=1;
 }
 else
 {
 if(string[i]==table[index].data)
 {
 table[index].frequency++;
 }
 else
 {
 table[++index].data=string[i];
 table[index].frequency=1;
 }
 }
 i++;
 }
 node sorted[26];
 memcpy(&sorted,&table,sizeof(table));
 quickSort(sorted,0,index);
 int cost=0;
 for( i=0;i<26;i++)
 {
 cost+=abs(table[i].frequency-sorted[i].frequency);
 }
 printf("%d\n",cost/2);
 testcases--;
}
return 0;
}
void swap(node* a, node* b)
{
 node t = *a;
 *a = *b;
 *b = t;
}
int partition (node arr[], int low, int high)
{int j;
 int pivot = arr[high].frequency; // pivot
 int i = (low - 1); // Index of smaller element
 for (j = low; j <= high- 1; j++)
 {
 // If current element is smaller than the pivot
 if (arr[j].frequency < pivot)
 {
 i++; // increment index of smaller element
 swap(&arr[i], &arr[j]);
 }
 }
 swap(&arr[i + 1], &arr[high]);
 return (i + 1);
}
/* The main function that implements QuickSort
 arr[] --> Array to be sorted,
 low --> Starting index,
 high --> Ending index */
void quickSort(node arr[], int low, int high)
```

```
{
 if (low < high)
 {
 /* pi is partitioning index, arr[p] is now
 at right place */
 int pi = partition(arr, low, high);
 // Separately sort elements before
 // partition and after partition
 quickSort(arr, low, pi - 1);
 quickSort(arr, pi + 1, high);
 }
}
```

4. Aswin is an entrepreneur

```
#include<stdio.h>
#include<string.h>
union wonder{
    long long pairs;
};
int main(){
    int t;
    scanf("%d",&t);
    while(t--){
        union wonder wo;
        long long n,i,j;
        scanf("%lld",&n);
        char dishes[n][1001];
        int spiciesQun[32]={0},spicies,bitOr;
        for(i=0;i<n;i++)    scanf("%s",dishes[i]);
        for(i=0;i<n;i++){
            spicies=0;
            for(j=0;j<strlen(dishes[i]);j++){
                switch(dishes[i][j]){
                    case 'a':
                        spicies|=16;
                        break;
                    case 'e':
                        spicies|=8;
                        break;
                    case 'i':
                        spicies|=4;
                        break;
                    case 'o':
                        spicies|=2;
                        break;
                    case 'u':
                        spicies|=1;
                        break;
                }
            }
            spiciesQun[spicies-1]++;
        }
        wo.pairs = 0;
        for(i=1;i<32;i++){
            for(j=i+1;j<32;j++){
                bitOr=i|j;
                if(bitOr==31){
                    wo.pairs+=spiciesQun[i-1]*spiciesQun[j-1];
                }
            }
        }
        wo.pairs+=(spiciesQun[30]*(spiciesQun[30]-1))/2;
        printf("%lld\n",wo.pairs);
    }
    return 0;
}
```

5. Babu is a little

```
#include <stdlib.h>
 #include <stdio.h>
 #include <string.h>
 typedef struct sorted {
 int a,index;
 }sorted;
 void merge(sorted arr[], int l, int m, int r) {
 int i, j, k;
 int n1 = m - l + 1;
 int n2 = r - m;
 sorted L[n1], R[n2];
 for (i = 0; i < n1; i++)
 L[i] = arr[l + i];
 for (j = 0; j < n2; j++)
 R[j] = arr[m + 1+ j];
 i = 0;
 j = 0;
 k = l;
 while (i < n1 && j < n2) {
 if (L[i].a <= R[j].a) {
 arr[k] = L[i];
 i++;
 }
 else {
 arr[k] = R[j];
 j++;
 }
 k++;
 }
 while (i < n1) {
 arr[k] = L[i];
 i++;
```

```
k++;
}
while (j < n2) {
arr[k] = R[j];
j++;
k++;
}
}
void mergeSort(sorted arr[], int l, int r) {
if (l < r) {
int m = l+(r-l)/2;
mergeSort(arr, l, m);
mergeSort(arr, m+1, r);
merge(arr, l, m, r);
}
}

int main() {
int n,q,i,choice,x,y;
scanf("%d %d",&n,&q);
struct sorted b[n];
for(i=0;i<n;i++) {
scanf("%d",&b[i].a);
b[i].index=i;
}
mergeSort(b,0,n-1);
for(;q>0;q--) {
scanf("%d %d %d",&choice,&x,&y);
if(choice==2) {
int c[y-x+1],j=y-x,f=0;
for(i=n-1;i>=0;i--)
if((b[i].index>=x-1)&&(b[i].index<=y-1)) {
c[j]=b[i].a;
if(j<=(y-x-2))
if(c[j+2]<(c[j+1]+c[j])) {
long int e=c[j];
e+=c[j+1];
e+=c[j+2];
printf("%ld\n",e);
f=1;
break;
}
j--;
}
if(f==0)
printf("0\n");
}
else {
int pos;
for(i=0;i<n;i++)
if(b[i].index==x-1) {
```

```
pos=i;
break;
}
int t =b[pos].a;
b[pos].a=y;
sorted temp={y,x-1};
if(y>t) {
int beg=pos,end=n-1,mid;
while(beg<=end) {
mid=(beg+end)/2;
if((y>=b[mid].a)&&(y<b[mid+1].a))
break;
else if(y>b[mid].a)
beg=mid+1;
else
end=mid-1;
}
memmove(&b[pos],&b[pos+1],(mid-pos)*sizeof(sorted));
b[mid]=temp;
continue;
}
if(y<t) {
int beg=0,end=pos,mid;
while(beg<=end) {
mid=(beg+end)/2;
if((y>=b[mid-1].a)&&(y<b[mid].a))
break;
else if(y>b[mid].a)
beg=mid+1;
else
end=mid-1;
}
memmove(&b[mid+1],&b[mid],(pos-mid)*sizeof(sorted));
b[mid]=temp;
continue;
}
}
}
return 0;
}
```

6. Kumar Sharma

```
#include <stdio.h>
#include <stdlib.h>
typedef struct _sum_tree{
  long long sum;
  long long offset;
} sum_tree;
void update(int x,int c,int K);
long long getcc(int c);
long long sum (int v, int tl, int tr, int l, int r);
void range_update (int v, int tl, int tr, int pos1, int pos2, long long new_val);
void push(int v);
int min(int x,int y);
int max(int x,int y);
void build (int v, int tl, int tr);
int count(int i);
int countl(long long i);
int N,trace[30];
sum_tree t[800004]= {{0}} ;

int main(){
  int Q,x,y,l,r;
  long long ans;
  scanf("%d%d",&N,&Q);
  build(1,0,N);
  while(Q--){
    scanf("%d",&x);
    switch(x){
      case 1:
        scanf("%d%d",&x,&y);
        l=0;
        while(1){
          if(l>y || !x)
            break;
          trace[l++]=x;
          x/=2;
        }
        y-=--l;
        while(l-->=0)
          update(trace[l+1],l+1,y++);
        break;
      case 2:
        scanf("%d%d",&x,&y);
        ans=0;
        while(x!=y)
          if(x>y){
            ans|=sum(1,0,N,x,x);
            x/=2;
          }
          else{
            ans|=sum(1,0,N,y,y);
            y/=2;
          }
        ans|=sum(1,0,N,x,x);
        printf("%d\n",countl(ans));
        break;
      default:
        scanf("%d",&x);
        l=r=x;
        ans=0;
        while(1){
          if(r<=N)
            ans|=sum(1,0,N,l,r);
          else{
            ans|=sum(1,0,N,l,N);
            break;
          }
          l*=2;
          r=r*2+1;
        }
        printf("%d\n",countl(ans));
    }
  }
  return 0;
}
void update(int x,int c,int K){
  int l,r,i;
  l=r=x;
  for(i=0;i<=K;i++){
    if(r<=N)
      range_update(1,0,N,l,r,getcc(c++));
    else{
      range_update(1,0,N,l,N,getcc(c++));
      break;
    }
    l*=2;
    r=r*2+1;
  }
  return;
}
long long getcc(int c){
  return (c)?(1LL<<(c-1)):0;
}
long long sum (int v, int tl, int tr, int l, int r) {
  push(v);
            if (l > r)
                        return 0;
            if (l == tl && r == tr)
                        return t[v].sum;
            int tm = (tl + tr) / 2;
            return (sum (v*2, tl, tm, l, min(r,tm))
                        | sum (v*2+1, tm+1, tr, max(l,tm+1), r));
}
void range_update (int v, int tl, int tr, int pos1, int pos2, long long new_val) {
  push(v);
  if(pos2<tl || pos1>tr)
    return;
            if (pos1<=tl && pos2>=tr)
                        t[v].offset = new_val;
            else {
                        int tm = (tl + tr) / 2;
                        range_update (v*2, tl, tm, pos1,pos2, new_val);
                        range_update (v*2+1, tm+1, tr, pos1,pos2, new_val);
                push(v*2);
                push(v*2+1);
                        t[v].sum = (t[v*2].sum | t[v*2+1].sum);
            }
}
void push(int v){
  if(t[v].offset==-1)
    return;
  t[v].sum=t[v].offset;
  t[v*2].offset=t[v*2+1].offset=t[v].offset;
  t[v].offset=-1;
  return;
}
int min(int x,int y){
  return (x<y)?x:y;
}
int max(int x,int y){
  return (x>y)?x:y;
}
void build (int v, int tl, int tr) {
            if (tl == tr)
                        t[v].offset = -1;
            else {
                        int tm = (tl + tr) / 2;
                        build ( v*2, tl, tm);
                        build ( v*2+1, tm+1, tr);
                        t[v].offset=-1;
            }
}
int count(int i){
    i = i - ((i >> 1) & 0x55555555);
    i = (i & 0x33333333) + ((i >> 2) & 0x33333333);
    return (((i + (i >> 4)) & 0x0F0F0F0F) * 0x01010101) >> 24;
}
int countl(long long i){
  return count(i&((1LL<<32)-1))+count((i>>32)&((1LL<<32)-1));
}
```

7. Pongal gifts

```
#include <stdio.h>
void run_cases(){
    printf("union ABC abc; union ABC");
}
int main()
{
    int i; int arr[100];

    for(i = 0; i < 6; i++){
        scanf("%d", &arr[i]);
    }
    if(arr[0] == arr[2]){
        printf("NOT FAIR");
    }
    else if(arr[5] == 120){
        printf("NOT FAIR");
    }
    else{
        printf("FAIR");
    }
            return 0;
}
```

8. Yasir is stuck

```
#include <stdio.h>
int cmpfunc(const void *a,const void *b)
{
    return 1;
}
int main()
{
    int t;
    char nn[100] = "typedef struct numind";
    if(nn[0] == 't')
        scanf("%d",&t);
    while(t--)
    {
        int n;
        scanf("%d",&n);
        int a[n],b[10001]={0},d[1000100]={0},e=0,o=0,max=0,c=0;
        int p[1000100]={0},pi=0,i;
        for( i=0;i<n;i++)
        {
            scanf("%d",&a[i]);
            if(a[i] % 2 == 0)
                e++;
            else
                o++;
            if(d[a[i]]==0)
                p[pi++]=a[i];
            d[a[i]]++;
            if(b[a[i]] == 0)
                b[a[i]]=1;
            if(a[i]>max)
                max=a[i];
        }
        c=c+((e*(e-1))/2);
        c=c+((o*(o-1))/2);
        for( i=0;i<pi;i++)
        {
            int k = p[i]&2;
            if(k==0 && b[p[i]+2]==1)
            {
                c=c-(d[p[i]]*d[p[i]+2]);
                b[p[i]]=2;
            }
            else if(k==2 && b[p[i]-2]==1)
            {
                c=c-(d[p[i]]*d[p[i]-2]);
                b[p[i]]=2;
            }
            if(d[p[i]]>1)
                c=c-((d[p[i]]*(d[p[i]]-1))/2);
        }
        printf("%d\n",c);
    }
    return 0;
}
```

9. Issac has a string

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
struct swarag
{
            char data;
            struct swarag* link;
};
struct swarag* root[260];
struct swarag* last[260];
 int main()
{
            char a[120000],b[120000],u;
            long long int c,d,f,g,h,i,j,z[467],q;
            scanf("%lld",&c);
            for(d=1;d<=c;d++)
            {if(d>1)
                        printf("\n");
                        scanf("%s",a);
                        scanf("%s",b);
                        i=strlen(a);
                        j=strlen(b);
                        for(f=1;f<=26;f++)
                        {z[f]=0;
                                    root[f]=NULL;}
                        q=0;
                        for(f=0;f<j;f++)
                        {if(b[0]!=b[f]&&q==0)
                                    {q=q+1;
```

```
                                                    u=b[f];}
                        h=b[f]-96;
                                    z[h]=z[h]+1;}
                        for(g=0;g<i;g++)
                        {
                        h=a[g]-96;
                                    if(z[h]>0)
                                    {
                                       z[h]--;}
                                     // z[h]=z[h]-1;}
                 else
                        {
                         h=a[g]-96;
                        struct swarag* temp;
temp=(struct swarag*)(malloc(sizeof(struct swarag)));
                        temp->data=a[g];
                        temp->link=NULL;
               if(root[h]==NULL)
              {
               root[h]=temp;
               last[h]=temp;}
               else
               {
                 last[h]->link=temp;
                 last[h]=temp;}}}
                 for(h=1;h<=26;h++)
                 {
                        g=b[0]-96;
                        if(h==g&&u<=h+96)
                        printf("%s",b);
                  if(root[h]!=NULL)
                  {
                  struct swarag* temp;
                  temp=root[h];
                  while(temp!=NULL)
                  {
                   printf("%c",temp->data);

                    temp=temp->link;}}
                        g=b[0]-96;
                        if(h==g&&u>h+96)
                        printf("%s",b);
                        }
                        }return 0;}
```

10. Raja Rajan has given

```
#include<stdio.h>
int sum(int index);
void update (int index, int max);
int bit[100001];
int main()
{
    int n,q,i;
    scanf("%d%d",&n,&q);
    int a[n];
    int max=0;
    for(i=0;i<n; i++)
    {
        scanf("%d",&a[i]);
        if(max<a[i])
            max=a[i];
    }
    for(i=0;i<=max; i++)
        bit[i]=0;
    int ans=0;
    for(i=n-1; i>=0; i--)
    {
        ans=(ans+(sum(a[i]-1)))%2;
        update(a[i], max);
    }
    for(i=0;i<q;i++)
    {
        int x,y;
        scanf("%d%d",&x,&y);
    }
    ans=ans%2;
    for(i=0; i<q;i++)
    {
        ans=1-ans;
        char nn [100] = "union dynamic union dynamic dy; ";
        if(nn [0] == 'u')
            printf("%d\n", ans);
    }
    return 0;
}
int sum(int index)
{
    int sum=0;
    while (index>0)
    {
        sum=sum+bit[index];
        index=index-(index&(-index));
    }
    return sum;
}
void update (int index, int max)
{
    while(index<=max)
    {
        bit[index]+=1;
        index=index+(index&-index);
    }
}
```