Week 6

Q1.

#include <algorithm > #include <iostream> #include <vect or> # include<stack> using namespace std;

int dfs(vect or<v ector<int>>a, int v,int n,int st[10],int end)

{

int l,x; stack<int>s; s.push(v);

while(!s.empty())

{

l=s.top();

if(l==(end -1)) return 0; st[l]=2;

s.pop();

for( int i=0;i<n ;i++)

{

if(a[l][i]==l)

{

if(st[i]!=2)

s.push(i);

st[i]=2; a[l][i]=2;

}

}

}

return 1;

}

int main() {

int c,i,j,n,key,st[ 10],start,end ; vector<v ector<int>>a ;

cin»n; for(i= 0;i<n ;i++)

{

vector<int>temp; for( j=0;j<n ;j++)

{

cin»key; temp.push\_back(key);

}

st[i]=0;

a.push\_back(temp);

}

cin»start»end; c=dfs(a,start-1,n,st,end);

if(c==0)

cout« "Path exist"; else

cout« "No path exist"; return 0;

}

# Output-

5

01100

10111

11010

01101

01010

15

**Q2.**

#include < bits /stdc++.h> using namespace std;

bool isBipartiteUtili(nt G[][10], int src, int colorArr[],int V)

{

colorArr[src] = 1; queue< int> q; q.push(src);

while (!q.empty()) { int u = q.front(); q.pop();

if (G[u][u] == 1)

return false;

for ( int v = 0; v < V; ++v) {

if (G[u][v] && colorArr[v] == -1) { colorArr[v] = 1 - colorArr[u]; q.push(v);

}

else if (G[u][v] && colorArr[v] return false;

}

}

colorArr[u])

return true;

}

bool isBipartite(int G[][10],int V)

{

int colorArr[VJ;

for ( int i = 0; i < v,· ++i) colorArr[i] = -1;

for ( int i = 0; i < v,· i++)

if (colorArr[i] -- -1)

if (isBipartiteUtil(G, i, colorArr,V) return false;

return true;

}

false)

int main()

{

int V,i,j;

int g[10][10]; cin»V; for(i= 0;i<V;i++)

{

for( j=0;j<V ;j++) cin»g[i][j];

}

isBipartite(g,V) ? cout « "Bipartite" cout « "Not Bipart ite";

**return 0 ;**

}

# Output-

**5**

**01100**

**10111**

**11010**

**01101**

**01010**

## Q3.

#include <algorithm >

#include <iostream>

#include <vect or> # include<queue > using namespace std;

typedef pair< int,int>p i;

bool dfs(int s,vector< bool> &v isited,vector<pa ir<int,int>>pq, int n) { if (visited[s]) return true;

visited[s] = 1;

// process nodes

for ( int i=0;i<n ;i++){

if( pq[i] .first==s && dfs(pq[i].second,visited,pq,n)) return true;

}

visited[s]= false; return false;

}

int cycle( vect or<pa ir<int,int>>pq, int n)

{

for( int i=l;i<=n ;i++){

vector< bool> visited(n +l,false);

if(dfs(i,visited,pq,n)){ return 1;

}

}

return 0;

}

int main() {

int co=0,c ,i,j,n,key,p[l0]; vector<pa ir<int,int>>pq ; cin»n;

for(i= 0;i<n ;i++)

{

for( j=0;j<n ;j++)

{

cin»key; if(key!=0)

{

pq.push\_back(make\_pair(i,j)); co++;

}

}

}

c=cycle(pq,co); if(c==0)

cout« "no cycle exist";

else

cout« "yes cycle exist"; return 0;

}

#### OUTPUT-

5

01100

00011

01010

00001

00000

## Week 7

## Q1.

#include<iostream> # include< bits /stdc++.h> using namespace std;

int minDisindex(int \*dis, bool \*vis,int v)

{

int i;

int minDis=INT\_MAX; int minindex=- 1; for(i= 0;i<v ;i++)

{

if(vis[i]== false && dis[i]<=minDis)

{

minDis=dis[i]; minindex=i;

}

}

return minindex;

}

void dijkstra(vector<vector< int>> mat,int v,int s)

{

int dis[v]; bool vis[v]; int parent[v]; int i,j;

for(i= 0;i<v ;i++)

{

**dis[i]=INT\_MAX;** vis[i]= false; parent[i]=- 1;

}

dis[ s]=0; parent[s]=s; for(i= 0;i<v ;i++)

{

int m=minDisindex(dis,vis,v); vis[m]= true;

for( j=0;j<v ;j++)

{

if(dis[m]!= INT\_MAX && !vis[j] && mat[m][j])

{

if(dis[ j]>d is[m]+mat[m][j])

{

dis[j]=dis[m]+mat[m][j]; parent[j]=m;

}

}

}

}

for(i= 0;i<v ;i++) { if(i==S) {

cout« i+l« 11

continue;

11 «dis[i]«endl;

}

cout« i+l; j=i;

while( parent[ j]!=s) {

cout« 11 11 « parent[j] +l; j=parent[j];

}

cout<< 11 11 << s+l<< 11 11 « dis[i]« endl;

}

}

int main()

{

int i,j; int v; cin»v;

vector<v ector<int>> mat(v,vect or<int> (v)); for(i= 0;i<v ;i++)

for( j=0;j<v ;j++) cin»mat[i][j]; int s;

cin»s;

dijk stra(mat,v,s-1); return 0;

}

### Output­

s

04100

00004

02040

00004

00000

1

1 0

2 3 1 3

3 1 1

4 3 1 5

5 2 3 1 7

**Q2.**

#include < bits /stdc++.h> using namespace std;

void calulate(vect or<int> &pa, int i)

{

cout « i + 1 « II II ,•

if (pa[i] >= 0) calulate(pa, pa[i]);

}

void find\_path( int \*\*graph, int m, int sour)

{

vector<int > dis(m, **INT\_MAX),** pa(m, -1); dis[sour] = 0·,

for ( int ki = 0; **ki** < m - 1; ki++)

{

for ( int i = 0; i < m·,

{

i++)

for ( int j = 0; j < m; j++)

{

if (graph[i][j] != 0)

{

if (dis[j] > dis[i] + graph[i][j])

{

dis[j] = dis[i] + graph[i][j]; pa[j] i·,

}

}

}

}

}

for ( int i = 0; i < m; i++)

{

calulate(pa, i);

cout « 11

•

}

}

11 « dis[i] « endl;

int main()

{

int m, source, ed; cin » m·,

int \*\*graph= ( int \*\*)malloc(m \* sizeof(int \*));

for ( int i = 0; i < m·, i++)

graph[i] = ( int \*)malloc(m \* sizeof(int));

for ( int i = 0·, i < m·, i++) {

for ( int j = 0; j < m; j++) {

cin » graph[i][j];

}

}

**cin » source;**

**find\_path(graph, m, source - 1);**

}

### Output­

**5**

**04100**

**00004**

**02040**

**00004**

**00000**

**1**

|  |  |  |  |
| --- | --- | --- | --- |
| 1 | 0 |  | |
| 2 3 | 1 |  | 3 |
| 3 1 |  | 1 |  |
| 4 3 | 1 |  | 5 |
| 5 2 | 3 | 1 | 7 |

## Q3.

#include < bits /stdc++.h> using namespace std;

int shortest\_weigti(nt \*\*graph, int ver, int u, int v, int k)

{

if (k <= 0)

return INT\_MAX; if (k == 0 && u == v)

return 0;

if (k == 1 && graph[u][v] != INT\_MAX) return graph[u][v];

int res= INT\_MAX;

for ( int i = 0; i < ver; i++) {

if (graph[ u][i] != 0 && u != i && v != i) {

int recu = shortest\_weigt(graph, ver, i, v, k - 1); if (recu != INT\_MAX)

res= min(res, graph[u][i] + recu);

}

}

return res;

}

int main()

{

int ver, u, v, k, ans; cin » ver;

int \*\*graph= ( int \*\*)malloc(ver \* sizeof(int \*)); for ( int i = 0; i < ver; i++)

graph[i] = ( int \*)malloc(sizeof(int) \* ver); for ( int i = 0; i < ver; i++)

for ( int j = 0; j < ver; j++) cin >> graph[i][j];

cin » **LI** » **V** » **k·**,

ans = shortest\_weigt(graph, ver, **LI** - **1 , V** - **1 , k);**

cout « "Weight of shortest path from ( II « **LI**

« « **V** « ") with II « k « II edges • II « ans;

}

### Output-

4

0 10 3 2

0 0 0 7

0 0 0 6

0 0 0 0

1 4

2

Weight of shortest path from (1,4) with 2 edges

## Week 8

## Q1.

#include < bits /stdc++.h> #define 11 long long #define INF INT MAX using namespace std;

int prims( int \*\*arr, int n)

{

vector< bool> visited(n, false ); vector<int> weight(n, INF);

priority\_queue<pair< int, int>, vector<pa ir<int, int>>, greater<pa ir<int, int» > min\_heap;

int src = 0; weight[src] = 0;

min\_heap.push(make\_pa ir(weight[src], src)); while (!min\_heap.empty())

{

int u = min\_heap.top().second; min\_heap. pop();

if (!visited[u])

{

visited[u] true;

for ( int v 0; v < n; ++v)

{

if (!visited[v] && arr[u][v] != 0 && arr[u][v] < weight[v])

{

weight[v] = arr[u][v]; min\_heap.push(make\_pair(weight[v], v));

}

}

}

}

int sum= 0;

for ( auto i : weight) sum+= i;

return sum;

}

int main()

{

int n; cin » n; int \*\*arr;

arr= (int \*\*)malloc(n \* sizeof(int \*));

for ( int i

arr[i]

0; i < n; ++i)

( int \*)malloc(n \* sizeof(int));

for ( int i = 0; i < n; ++i) for ( int j = 0; j < n; ++j)

**cin >> arr[i][j];**

**cout « "Minimum spanning weight return 0;**

}

" « **prims(arr, n);**

#### Output-

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **7** |  | | | | | |
| **0** | **0** | **7 5** | | **0** | **0** | **0** |
| **0** | **0** | **8 5** | | **0** | **0** | **0** |
| **7**  **5** | **8**  **5** | **0 9**  **9 0** | | **7 0**  **15** | | **0**  **6 0** |
| **0** | **0** | **7** | **15 0 8 9** | | | |
| **0** | **0** | **0** | 6 **8 0 11** | | | |
| **0** | **0** | **0** | **0 9 11 0** | | | |

Minimum spanning weight : 39

#### Q2.

#include <algorithm > #include <iostream> #include <vect or> # include<queue > using namespace std;

typedef pair< int,pair<int,int>>p i; int find( int p[l0],int x)

{

if(p[x]== -1) return x;

return find(p,p[x]);

}

void unionl(int u,int v,int s[10],int p[10])// here sfor parent and p for status

{

int pu=find(s,u); int pv=find(s,v); if(pu!=pv)

{

if( p[pu]<p[pv])

{

s[pu]=pv; p[pv]=p[pv]+p[pu];

}

else

{

s[pv]=pu; p[pu]=p[pu]+p[pv];

}

}

}

void kruskal( priority\_queue<pi,vec tor<p i>, greater<p i>>pq, int n,int p[l0],int st[10])

{

int sum=0,i,j,w,pi,p j;

while(!pq.empty())

{

w=pq.top().first; i=pq.top().second.first; j=pq.top().second.second;

pq.pop();

pi=find(p,i);

pj=find(p,j); if(pi!=pj )

{

unionl(i,j,p,st); sum+=w;

}

}

cout«sum;} int main() {

int c=0,i,j,n,key,p[ l0],st[10]; priority\_queue<pi,vector<pi>,greater<pi>>pq; cin»n;

for(i= 0;i<n ;i++)

{

for( j=0;j<n ;j++)

{

cin»key; if(key!=0)

{

pq.push(make\_pair(key,make\_pair(i,j))); c++;

}

st[i]=l;

**p[ i ] =- 1 ;**

}

}

kruskal(pq,c,p,st); return 0;

}

### Output-

7

0075000

0085000

7809700

55901560

00715089

00068011

00009110

39

## Q3.

#include <algorithm > #include <iostream> #include <vect or> # include<queue > using namespace std;

typedef pair< int,pair<int,int>>p i; int find( int p[l0],int x)

{

if(p[x]== -1) return x;

return find(p,p[x]);

}

void unionl(int u,int v,int s[10],int p[10])// here sfor parent and p for status

{

int pu=find(s,u); int pv=find(s,v); if(pu!=pv)

{

if( p[pu]<p[pv])

{

}

else

{

s[pu]=pv; p[pv]=p[pv]+p[pu];

s[pv]=pu; p[pu]=p[pu]+p[pv];

} }}

void kruskal( priority\_queue<pi,vec tor<p i>>pq, int n,int p[l0],int st[10])

{

int sum=0,i,j,w,pi,p j; while(!pq.empty())

{

w=pq.top().first; i=pq.top().second.first; j=pq.top().second.second; pq.pop();

pi=find(p,i);

pj=find(p,j); if(pi!=pj )

{

unionl(i,j,p,st); sum+=w;

}

}

cout«sum;} int main() {

int c=0,i, j,n,key,p[10],st1[0]; priority\_queue<pi,vector<pi>>pq; cin»n;

for(i= 0;i<n ;i++)

{

for( j=0;j<n ;j++)

{

cin»key; if(key!=0)

{

pq.push(make\_pair(key,make\_pair(i,j))); c++;

}

st[i]=l;

**p[ i ] =- 1 ;**

}

}

kruskal(pq,c,p,st); return 0;

}

#### Output-

7

0075000

0085000

7809700

559 0 15 6 0

0 0 7 15 0 8 9

0 0 0 6 8 0 11

00009110

**59**

##### Week 9

##### Q1.

#include < bits /stdc++.h> using namespace std;

int main()

{

int n, i, j, k, w;

cin » n;

int graph[n][n]; string temp;

for (i = 0; i < n; i++)

{

for (j = 0; j < n; j++)

{

cin » temp;

if (temp != "INF")

{

graph[i][j] stoi(temp);

} else {

graph[i][j] le8;

}

}

}

for (k = 0; k < n; k++)

{

for (i = 0; i < n; i++)

{

for (j = 0; j < n; j++)

{

if (graph[i][k] + graph[k][j] < graph[i][j])

{

graph[i][j] = graph[i][k] + graph[k][j];

}

}

}

}

cout « "The shortest path matrix: " « endl; for (i = 0; i < n; i++)

{

for (j = 0; j < n; j++)

{

if(graph[i][j] >= le8) cout « "INF"; else cout << graph[i][j];

cout << " ";

}

cout << endl;

}

**return 0 ;**

}

**OUTPUT-**

**5**

**0 10 5 5 INF**

**INFO 5 5 5**

**INF INFO INF 10 INF INF INFO 20 INF INF INF 5 0**

The shortest path matrix:

0 10 5 5 15

INFO 5 5 5

INF INFO 15 10 INF INF INFO 20 INF INF INF 5 0

###### Q2.

#include < bits /stdc++.h> using namespace std;

int main()

{

int n; cin » n;

vector<double > items(n); vector<double > val(n); vector<v ector<double >> job; for ( int i = 0; i < n; i++)

{

cin » items[i];

}

for ( int i = 0; i < n; i++)

{

cin » val[i];

job.push\_back({val[i] / items[i], items[i], ( double )(i + 1)});

}

double k; cin » k;

sort(job.rbegin(), job.rend()); vector<pa ir<double , double >> ls; float profit= 0;

for ( int i = 0; i < n; i++)

{

if ( job[i][l] >= k)

{

profit+= k \* job[i][0]; ls.push\_back(make\_pair(k, job[i][ 2])); break ;

}

else

{

profit+= job[i][ l] \* job[i][0];

}

ls.push\_back( mak e\_pair(job[i][ l], job[i][ 2])); k = k - job[i][ l];

}

cout « "Maximum Value · " « profit « endl; cout « "Item - Weight" « endl;

for ( auto it : ls)

cout « it.second « " - " « it.first « endl; return 0;

}

###### OUTPUT-

**6**

**610 3 5 13**

**621835**

**16**

Maximum Value : 22.3333

Item - Weight

5 - 1

6 - 3

4 - 5

1 - 6

3 - 1

###### Q3.

#include < bits /stdc++.h> using namespace std;

int main()

{

int n; cin » n;

vector<int> a(n);

for ( int i = 0; i < n; i++)

{

cin » a[i];

}

priority\_queue< int, vect or<int>, greater<int>> minheap; for ( int i = 0; i < n; i++) {

minheap.push(a[i]);

}

int ans= 0;

while (minheap.size() >1)

{

int el= minheap.top(); minheap.pop();

int e2 = minheap.top(); minheap.pop();

ans+= el+ e2; minheap.push(el + e2);

}

cout « ans; return 0;

}

###### OUTPUT-

10

10 5 100 50 20 15 5 20 100 10

895

##### Week 10

##### Q1.

#include< bits /stdc++.h> using namespace std; int main() {

int n; cin»n;

int i,s[n],f[n]; for(i=0;i<n ;i++) cin»s[i]; for(i= 0;i<n ;i++) cin»f[i]; vector<v ector<int>> a; vector<int > act; for(i= 0;i<n ;i++)

a.push\_back({ f[i],s[i],li}+);

sort(a.begin(),a.end()); int **e=INT\_MIN,c= 0;** for(i= 0;i<n ;i++)

{

if(a[i][ l]>= e)

{

e=a[i][0]; c++;

act.push\_back(a[i][ 2]);

}

}

cout« 11 No. of non-conflicting activities 11 « c« endl;

cout« 11 List of selected activities for(i= 0;i<ac t.size();i++) cout«act[i]« 11 11

, ;

return 0;

}

II ,•

###### OUTPUT-

10

1 3 0 5 3 5 8 8 2 12

4 5 6 7 9 9 1112 1416

No. of non-conflicting activities : 4

List of selected activities : 1,4,7,10

###### Q2.

#include< bits /stdc++.h> using namespace std; int main()

{

int n; cin»n;

int i,t[n],f[n]; for(i= 0;i<n ;i++) cin»t[i]; for(i= 0;i<n ;i++) cin»f[i]; vector<v ector<int>> a; vector<int > act; for(i= 0;i<n ;i++)

a.push\_back({ f[i],f[i-]t[i],i+l}); sort(a.begin(),a.end());

int e=INT\_MIN,c=0; for(i= 0;i<n ;i++)

{

if(a[i][ l]>= e)

{

e=a[i][0]; c++;

act.push\_back(a[ i][2]);

}

}

sort(act.begin(),act.end());

cout«"Max number of tasks : "« c« endl; cout« "Selected task Numbers "; for(i= 0;i<ac t.size();i++)

cout« act[i]« ","; return 0;

}

###### OUTPUT-

7

2132221

2386253

Max number of tasks : 4

Selected task Numbers : 1,2,3,6

###### Q3.

#include< bits /stdc++.h> using namespace std; int main()

{

int n; cin»n;

int i,a[n],c,j; for(i= 0;i<n ;i++) cin»a[i];

bool f=0; sort(a,a+n); for(i= 0;i<n ;i++)

{

c=l; j=i+l;

while( j<n && a[j++]==a[i])

c++;

if( e>n/2)

{

cout« "yes\n"; f=l;

break ;

}

i=j-1;

}

if(f==0) cout<<"no\n"; if(n% 2!=0) cout« a[n/2]; else

cout<<((float)a[n/2]+a[n/2-1])/2; return 0;

}

###### OUTPUT-

9

442322322

yes

2

##### Week 11

##### Q1.

#include< bits /stdc++.h> using namespace std;

long matChainOrder(int \*p,int n) { int m[n][n];

int i,j,k,l,q;

for(i= l;i<n ;i++) m[i][i]=0; for(l= 2;l<n ;l++)

{

for(i= l;i<n- l+l;i++)

{

j=i+l-1; m[i][j]=INT\_MAX;

for(k=i;k <= j-l;k++)

{

q=m[i][k] +m[k +l][j]+p[i- l]\*p[k] \*p[j]; if(q<m[i][j])

m[i][j]=q;

}

}

}

return m[l][n-1];

}

int main()

{

int n; cin»n;

int p[n+l];

for( int i=0;i<n ;i++)

{

cin>>p[i]>>p[i+ l];

}

cout<<matChainOrder( p,n+l); return 0;

}

###### OUTPUT-

3

10 30

30 5

5 60

4500

###### Q2.

#include< bits /stdc++.h> using namespace std; int main()

{

int n,amt; cin»n;

int i,j,a[n]; for(i= 0;i<n ;i++) cin»a[i]; cin»amt;

int ans[amt+l]; for(i= l;i<=am t;i++) ans[i]= 0;

ans[0]=1; for( j=0;j<n ;j++)

{

for(i= l;i<=am t;i++)

{

if(a[ j]<= i) ans[i]+=(ans[i-a[j]]);

}

}

cout«ans[amt]; return 0;

}

###### OUTPUT-

4

2563

10

5

###### Q3.

#include< bits /stdc++.h> using namespace std; int main() {

int n; cin»n;

int i,j,a[n]; for(i= 0;i<n ;i++) cin»a[i];

int sum=0; for(i= 0;i<n ;i++) sum+=a[i]; if(sum% 2!=0)

{ cout« "no";

return 0;

}

sum=sum/2;

bool s[n+l][sum+l]; for(i= 0;i<=n ;i++)

{for( j=0;j<= sum;j++)

{

if( j==0) s[i][j]=l; else if(i==0) s[i][j]=0; else

{

if(a[i -l]>j) s[i][j]=s[i- l][j]; else

s[i][j]=(s[i- l][j] I I s[i-l][j-a[ i-1]]);} } }

if(s[n][sum]) cout«"yes"; else cout<<"no"; return 0;

}

###### OUTPUT-

7

15411514 10

Yes