به نام خدا

« حل مسئله ی رنگ آمیزی گراف به عنوان یک مسئله CSP با استفاده از الگوریتم عقبگرد »

درس هوش مصنوعی و سیستم های خبره

زیرنظر استاد محترم : دکتر محمد حسین خسروی

ارائه دهنده: احسان بیکی

تیرماه ۱۴۰۱

مستندات پیاده سازی مسئله ی رنگ آمیزی گراف با استفاده از الگوریتم عقبگرد و هیوریسستیک های گفته شده در درس و نیز سازگاری کمان

(برنامه نویسی شده به زبان c++)

```
#include <iostream>
#include <vector>
#include <list>
#include <bits/stdc++.h>
#include <queue>
#include <iomanip>
using namespace std;
struct arc
{
int xi;
int xj;
};
```

در نظر گرفتن یک ساختار از نوع کمان با جزء داده ای های xi و xj برای کمانxj-→xi

```
int n,c; // درنظر گرفتن دو متغیر برای ذخیره سازی تعداد متغیرها ( شهر ها یا استان ها) و تعداد رنگ ها
vector<string> Color; // درنظر گرفتن یک وکتور برای ذخیره سازی نام رنگ های دامنه ی مسئله
vector<int>ValuesLimit; // درنظر گرفتن یک وکتور برای نگهداری تعداد مقادیر باقی مانده از دامنه ی متغیرها
vector<bool> Selection; // درنظر گرفتن یک وکتور برای نگهداری وضعیت انتخاب متغیر ها (در صورت انتخاب یک متغیر، عنصر متناظر برابر ۱ می شود)
```

```
bool RecursiveBacktracking(int assignment[],bool **W,vector <list<int>>Domain); // تابع جست جو عقبگرد بازگشتی
int  SelectUnassignedVariable(int assignment[],bool **); // تابع انتخاب متغیرهای مقدار دهی نشده
bool RemoveInconsistent(arc x,vector < list <int> > Domain); // تابع حذف مقادیر ناسازگار از دامنه ی متغیرها
void ArcConsistency(bool **W,vector < list <int> > OriginalDomain); // تابع بررسی سازگاری کمان
void MRVChecking(bool signal,int var,bool **W); // MRV تابع بررسی و محاسبه تعداد مقادیر مجاز دامنه
bool Consistent(int var,int value,int assignment[],bool **W); // تابع بررسی سازگاری مقدار داده شده به یک متغیر با سایر متغیرها
bool BacktrackingSearch(bool **W,int Solotion[],vector < list <int> > Domain); // تابع جستجوی عقبگرد
int  MinRV(); // MRV تابع محاسبه مقدار با حداقل باقیمانده
```

```
bool RemoveInconsistent(arc x,vector < list <int> > Domain) // تابع حذف مقادیر ناسازگار از دامنه ی متغیرها
{
bool removed=false,condition; // درنظر گرفتن دو متغیر  برای وضعیت حذف مقادیر ناسازگار

for (list<int>::iterator xit=Domain[x.xi].begin() ; xit!=Domain[x.xi].end() ; xit++)
{
for (list<int>::iterator xjt=Domain[x.xj].begin() ; xjt!=Domain[x.xj].end() ; xjt++)
   if (*xit!=*xjt)
   {
     condition=0;
     break;
    }
   else
     condition=1;
```

در این بخش، برنامه بررسی می کند که برای یک مقدار مشخص از دامنه ی xi ، اگر مقداری در دامنه ی متغیر xj وجود داشته باشد که با آن مقدار(xi) متفاوت باشد، متغیر condition را برابر ۰ می کند و بررسی را متوقف می کند. در غیر این صورت، مقدار condition را برابر ۱ می کند و به بررسی ادامه می دهد.

```
 if(condition)
 {
    Domain[x.xi].remove(*xit);
     removed=true;
 }
}
```

در نهایت اگر مقدار متغیر condition برابر ۱ باشد، آن مقدار ناسازگار از دامنه ی متغیرxi حذف می شود و متغیر removed برابر true می شود و در ادامه سایر مقادیر دامنه ی xi بررسی می شوند.

```
    return removed; // مقدار removed برگشت داده می شود
}

void ArcConsistency(bool **W,vector < list <int> > OriginalDomain) ) // تابع بررسی سازگاری کمان
{
queue<arc> arcs_queue; // درنظر گرفتن یک صف از نوع داده ی کمان برای نگهداری کمان ها
for(int i=0;i<n;i++)
 for(int j=0;j<n;j++)
   if(W[i][j])
   {
     arc XIJ;
     XIJ.xi=i;
     XIJ.xj=j;
     arcs_queue.push(XIJ);
   }
```

پس از اجرای این بخش، برنامه تمام متغیرهایی را که مجاور یکدیگر هستند،به صورت یک کمان در نظر می گیرد و کمان ها را یکی یکی درون صف قرار می دهد.

در نظر گرفتن وکتوری از لیست پیوندی ها برای ذخیره کردن مقادیر دامنه ی متغیر ها // vector < list <int> > Domain;

تغییر سایز وکتور دامنه به تعداد متغیرها // Domain.resize(n);

```
for(int i=0;i<n;i++)
  for (list<int>::iterator it=OriginalDomain[i].begin() ; it!=OriginalDomain[i].end() ; it++)
    Domain[i].push_back(*it) ;
```

ایجاد یه نسخه کپی از مقادیر دامنه ها

تکرار حلقه تا زمانی که صف کمان ها خالی نشده باشد // while(arcs_queue.size()>0)

```
{
  arc x=arcs_queue.front();
  arcs_queue.pop();
```

در این بخش،برنامه عنصر ابتدای صف کمان ها را درکمان x ذخیره می گیرد و آن عنصر را از درون صف حذف می کند.

```
  if(RemoveInconsistent(x,Domain))
  {
    for(int k=0;k<n;k++)
      {
      if(W[k][x.xi])
        {
          arc NewX ;
          NewX.xi=k;
          NewX.xj=x.xi;
          arcs_queue.push(NewX);
       } } }}
}
```

اگر هنگام بررسی سازگاری کمان x ، مقداری ناسازگار از دامنه ی متغیر xi حذف شود، برنامه در یک حلقه ی for برای تمام متغیر های k که با متغیر xi مجاورهستند، کمان k→xi را برای بررسی سازگاری درون صف قرار می دهد.

تابع بررسی و محاسبه تعداد مقادیر مجاز دامنه MRV // void MRVChecking(bool signal,int var,bool **W)

```
{
        if(!signal)
        {
        for(int j=0;j<n;j++)
        {
                if (W[var][j]==1)
                {
                        --ValuesLimit[j];
                }
                if(ValuesLimit[j]<0)
                        ValuesLimit[j]=0;
        }

}
```

در صورتی که مقدار متغیر signal برابر ۰ باشد،(کاهنده)برنامه در یک حلقه ی for، به ازای تمام متغیرهای j مجاور با متغیر var، از تعداد مقادیر ممکن برای دامنه ی متغیرj،یک واحد کم می کند و اگر در نهایت تعداد مقادیر ممکن منفی شد، آن را برابر ۰ می کند.

```
        if(signal)
        {
        for(int j=0;j<n;j++)
        {
                if (W[var][j]==1)
                {
                        ++ValuesLimit[j];
                }
        }
        }
}
```

در صورتی که مقدار متغیر **signal** برابر ۱ باشد، (افزاینده)برنامه در یک حلقه ی **for**، به ازای تمام متغیرهای **j** مجاور با متغیر **var**، به تعداد مقادیر ممکن برای دامنه ی متغیر **j**، یک واحد اضافه می کند.

bool Consistent(int var,int value,int assignment[],bool **W) // تابع بررسی سازگاری مقدار داده شده به یک متغیر با سایر متغیرها

```
{
  for (int i=0;i<n;i++)
  {
    if(W[var][i]==1 && assignment[i]==value)
        return false;
   }
return true;
}
```

در این قسمت، برنامه برای تمام متغیر های مجاور متغیر **var** بررسی می کند در صورتی که مقدار تخصیص داده شده به آن متغیر ها با مقدار داده شده به متغیر **var** یکسان باشد،مقدار **false** را بر میگرداند و در غیر این صورت مقدار **true** را بر می گرداند.

bool BacktrackingSearch(bool **W,int Solotion[],vector < list <int> > Domain) // تابع جستجوی عقبگرد

```
{
  for(int i=0;i<n;i++)
    {
    for(int j = 0; j <c; j++)
        Domain[i].push_back(j) ;
    }
```

در این بخش، برنامه به ازای هر متغیر **i**، تمام مقادیر دامنه ی مسئله را به ترتیب، در عنصر **i** ام وکتور **domin** قرار می دهد.(مقدار دامین را با تمام رنگ ها مقدار دهی اولیه میکند.)

ArcConsistency(W,Domain); // فراخوانی تابع بررسی سازگاری کمان

int assignment[n]; // درنظرگرفتن یک آرایه به تعداد متغیر ها برای ذخیره ی مقادیر منتسب شده به متغیرها

```
for (int i=0;i<n;i++)
    assignment[i]=99;
```

در این حلقه، برای تمام متغیر ها، مقدار اولیه ی ۹۹ برا ی انتساب در نظر گرفته می شود. (این مقدار را به عنوان **Error** در نظر گرفته ایم یعنی به معنای انتساب همراه با خطاست)

bool result=false; // مقداردهی اولیه متغیر result با مقدار false

result = RecursiveBacktracking(assignment,W,Domain); // فراخوانی تابع عقبگرد بازگشتی

```
for (int i=0;i<n;i++)
  Solotion[i]=assignment[i];
```

در این حلقه، تمام عناصر آرایه ی assignment را به عنوان راه حل درون آرایه ی solotion قرار می دهیم.

```
return result; //     مقدار result برگشت داده می شود
}

bool RecursiveBacktracking(int assignment[],bool **W,vector < list <int> > Domain) //  تابع جست جو عقبگرد بازگشتی
{
   bool Complete=1; // ۱ با complete  مقداردهی اولیه متغیر
        for(int i=0;i<n;i++)
        {
          if(Selection[i]==0)
          {
            Complete=0;
            break;
            }
        }
        if (Complete==1)
              return true;
```

در این حلقه، برنامه برای تمام متغیرها، مقدار عنصر متناظر با آن را در آرایه ی selection بررسی می کند و اگر مقدار این عنصر برای متغیری برابر ۰ شد(انتساب کامل نباشد)، آنگاه متغیر complete برابر ۰ می شود و از حلقه خارج می شود.

اگر مقدار متغیر complete برابر ۱ باشد (انتساب کامل باشد)، مقدار true برگشت داده می شود.

```
    int var = SelectUnassignedVariable(assignment,W); //var  فراخوانی تابع انتخاب متغیرهای مقدار دهی نشده و مقدار دهی به
    int result=true; //  true با مقدار result مقداردهی متغیر

for (list<int>::iterator it=Domain[var].begin() ; it!=Domain[var].end() ; it++) // var تکرار حلقه به اندازه ی تعداد مقادیر دامنه ی متغیر
{
  int value= *it; // var  با یک مقدار از دامنه ی متغیر value مقداردهی متغیر

  if (Consistent(var,value,assignment,W))
  {  assignment[var]=value;
     MRVChecking(0,var,W);
```

اگر مقدار value در نظر گرفته شده برای متغیر var با مقادیر منتسب شده به سایر متغیرها سازگار باشد، همان مقدار value به متغیر var منتسب می شود. سپس، تابع بررسی مقادیر مجاز MRV با سیگنال ۰ (کاهنده)فراخوانی می شود.

```
result =RecursiveBacktracking(assignment,W,Domain); // فراخوانی تابع عقبگرد بازگشتی
if (result != false)
        return result;
else
{
MRVChecking(1,var,W);
for(int i=var+1;i<n;i++)
Selection[i]=0; }}
  }
return false; }     //مقدار false برگشت داده می شود
```

اگر مقدار result برابر true باشد(به یک انتساب کامل رسیده باشیم)،مقدار result برگشت داده می شود.

در غیر این صورت، تابع بررسی مقادیر مجاز دامنه VAR با سیگنال ۱(افزاینده) فراخوانی می شود. سپس در یک حلقه ی for، مقدار تمام عناصر آرایه ی selection برابر ۰ می شود.

```
int MinRV() // MRV تابع محاسبه مقدار با حداقل باقیمانده
{
int min=0 ;  // مقداردهی اولیه متغیر min با مقدار ۰
  for(int i=0;i<n;i++)
    {
      if(ValuesLimit[i]<ValuesLimit[min])
        min=i;
    }
        return min; //مقدار min برگشت داده می شود
}
```

با اجرای این حلقه، اندیس متغیری که کمترین تعداد باقیمانده ی مقادیر را دارد، به دست می آید و درون متغیر min قرار می گیرد.

```
int SelectUnassignedVariable(int assignment[],bool **W) // تابع انتخاب متغیرهای مقدار دهی نشده
{
int min; // در نظر گرفتن یک متغیر برای نگهداری متغیر با کمترین مقدار باقیمانده
for(int i=0;i<n;i++) // تکرار حلقه به اندازه ی تعداد متغیر ها
{
  min=MinRV(); //MRV  فراخوانی تابع محاسبه مقدار با حداقل باقیمانده جهت پیاده سازی هیوریستیک
  if(Selection[min]==0)  //در صورتی که متغیر با اندیس min انتخاب نشده باشد شرط if برقرار می شود
  {
    bool Condition=false; // مقداردهی اولیه متغیر condition با مقدار false

    for(int c=0;c<n;c++)
    {
      if(ValuesLimit[min]==ValuesLimit[c])
      {
        Condition=true;
        break;
      }
  }
```

بررسی انتخاب بین هیوریستیک MRV و هیوریستیک درجه

این حلقه، اگر تعداد مقادیر ممکن برای متغیر min با تعداد مقادیر ممکن برای متغیری دیگر برابر باشد، برنامه هیوریستیک درجه را مد نظر قرار میدهد.

```
if(Condition)
{
  int degree[n];
  for(int j=0;j<n;j++)
  {
    degree[j]=0;
    for(int k=0;k<n;k++)
    if (W[j][k]==1)
      ++degree[j];
  }
    int max=0;
    for(int d=0;d<n;d++)
    {
      if(degree[max]<degree[d])
      max=d;
    }
    if(Selection[max]==0)
    {
      Selection[max]=1;
      return max;
    }
}
```

هیوریستیک درجه

در صورتی که **condition** برابر **true** شود، آرایه ای به تعداد متغیرها برای ذخیره ی درجه ی محدودیت هر متغیر در نظر گرفته می شود .سپس در یک حلقه ی **for** ، در ابتدا عنصر **j** آرایه با ۰ مقداردهی می شود. در ادامه برحسب تعداد متغیرهای مجاور با متغیر موردنظر،درجه ی آن متغیر محاسبه می شود.

سپس متغیر **max** با مقدار ۰ ، مقداردهی اولیه می شود .در ادامه با استفاده از یک حلقه ی **for** ، عنصری از آرایه که بیشترین مقدار را دارد (بیشترین درجه) به دست می آید و اندیس آن در متغیر **max** ذخیره می شود.

در صورتی که متغیر با اندیس **max** انتخاب نشده باشد، آن را انتخاب می کند و **max** را بر می گرداند.

```
 Selection[min]=1;
 return min;
}
else if( Selection[i]==0)
 {
 Selection[i]=1;
 return i;
 }
 }
}
```

در صورتی که **condition** برابر **false** باشد ، متغیر با اندیس **min** انتخاب می شود و **min** برگشت داده می شود. (هیوریستیک **MRV**)

درغیر این صورت (متغیر با اندیس **min** انتخاب نشده باشد)، متغیر با اندیس **i** انتخاب و اندیس **i** برگشت داده می شود.

```
int main()
{
        vector < list <int> > Domain; // درنظر گرفتن یک وکتور از نوع داده ی لیست پیوندی برای ذخیره ی مقادیر دامنه ی متغیر ها س

        cout <<"Plz Enter The Number of Color: ";
        cin >> c;                                                    دریافت تعدادرنگ ها

        Color.resize(c); //  تغییر سایز وکتور color به تعداد رنگ ها

        cout <<"Plz Enter The Name of Colors: ";
        for(int i = 0; i <c; i++)                                    دریافت نام رنگ ها
                cin >> Color[i];

        cout <<"Plz Enter The Number of Variables: ";
        cin >> n;                                                    دریافت تعداد متغیرها

        Domain.resize(n);   // تغییر سایز وکتور Domain به تعداد متغیرها

        ValuesLimit.reserve(n);   // تغییر سایز وکتور ValuesLimit به تعداد متغیرها

        for(int i=0;i<n;i++)                                         برای تمام عناصر ValuesLimit ، تعداد رنگ ها به عنو ان مقدار اولیه ذخیره می شود.
                ValuesLimit[i]=c;

        Selection.resize(n);        // تغییر سایز وکتور Selection به تعداد متغیرها

        for(int i=0;i<n;i++)                                         تمام عناصر Selection ، با ۰ مقداردهی می شوند.
                Selection[i]=0;

        bool **W;                                                    درنظرگرفتن یک آرایه ی دو بعدی از نوع bool برای ذخیره ی داده های ماتریس همجواری
        W = new bool *[n];                                           (یک بودن هر درایه به معنای همسایگی سطر و ستون متناظر است)
        for(int i = 0; i <n; i++)
                W[i] = new bool[n];

        for(int i=0;i<n;i++)                                         در این حلقه، تمام عناصر آرایه ی W با ۰ مقداردهی اولیه می شوند.
                for (int j=0;j<n;j++)
                        W[i][j]=0;
```

```
string Variables[n];   // در نظرگرفتن آرایه ای به تعداد متغیرها برای ذخیره سازی نام متغیرها
cout <<"\nPlz Enter The Name Of Variables: ";
for (int i=0;i<n;i++)
        cin >> Variables[i];
```

دریافت نام متغیرها

```
cout <<"\nThe Variables Are: \n";
for (int i=0;i<n;i++)
        cout << Variables[i]<<"="<<i<<"    ";
cout <<"\n\n";
```

نمایش متغیرها به همراه اندیس اختصاص یافته به متغیر

```
for(int i=0;i<n;i++)
{
cout<<"\nPlz Enter The Neighbors Of Variable \"" << Variables[i] <<"(="<<i<<")"<< "\"(Zero Or one) :";
for (int j=0;j<n;j++)
        {
        cin >> W[i][j];
        }
}
```

دریافت داده های ماتریس همجواری تمام متغیرها

```
for(int i=0;i<n;i++)
{
cout<<"\nThe Neighbors Of Variable \"" << Variables[i] <<"\"(="<<i<<")"<< "\""<<setw(5)<<"\tis :"<<setw(10);
        for (int j=0;j<n;j++)
        {
                if(W[i][j])
                cout << Variables[j]<<setw(5);
        }
}
```

نمایش همسایه های تمام متغیرها جهت وضعیت صحت داده های ورودی

```
int Solotion[n]; // در نظرگرفتن آرایه ای به تعداد متغیرها  برای نگهداری راه حل (انتساب کامل و سازگار)

if(BacktrackingSearch(W,Solotion,Domain))
{
        cout <<"\n\n*********************************\n";
        cout <<"\n\nThe Solution is : \n";
        for (int i=0;i<n;i++)
        cout <<"\n"<< Variables[i]<<":"<<Color[Solotion[i]]<<"    ";
}
else
        cout <<"\n\nNo Solution Found!\n";

return 0;}
```

در صورتی که تابع BacktrackingSearch مقدار true برگرداند (مسئله به جواب رسیده باشد)،برنامه، راه حل مسئله را نمایش می دهد. در غیر این صورت، پیغام No Solution Found! را نمایش می دهد.

بررسی عملکرد برنامه با سه گروه از داده های ورودی

(شهر های استان خراسان جنوبی،استان های ایران،ایالات استرالیا)

نمونه اول :

شهر های استان خراسان جنوبی:

تعداد رنگ ها : ۳

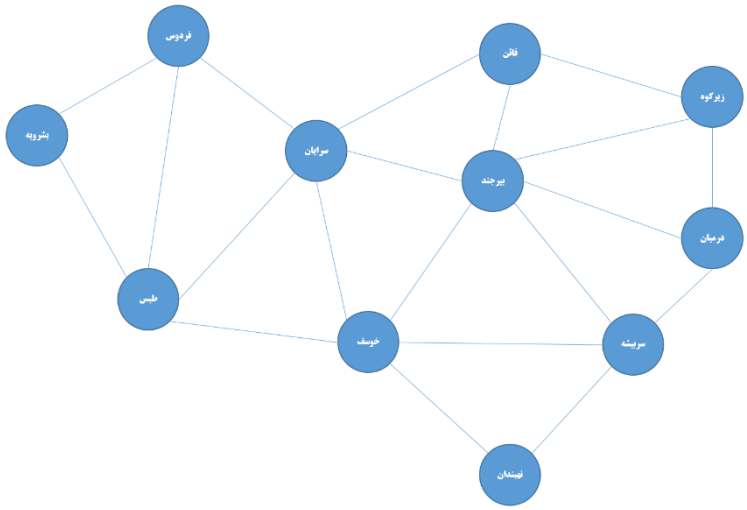
نام رنگ ها : Red,Green,Blue

تعداد متغیر ها: ۱۱

نام متغیر ها:

TB: Tabas **BSH**: Boshroyeh **FD**: Ferdos **SR**: Sarayan **KH**: Khoosf **GH**: Ghaenat
BJ: Birjand **ZK**: Zirkooh **DM**: Darmyan **SB**: Sarbisheh **NB**: Nehbandan

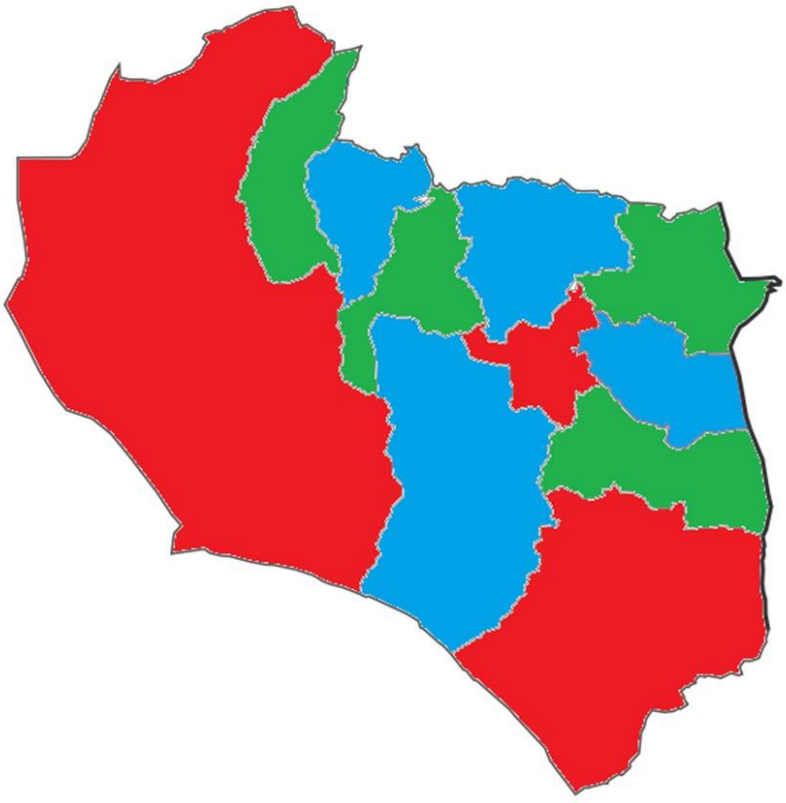
گراف:



ماتریس همجواری:

	TB	BSH	FD	SR	KH	GH	BJ	ZK	DM	SB	NB
TB	0	1	1	1	1	0	0	0	0	0	0
BSH	1	0	1	0	0	0	0	0	0	0	0
FD	1	1	0	1	0	0	0	0	0	0	0
SR	1	0	1	0	1	1	1	0	0	0	0
KH	1	0	0	1	0	0	1	0	0	1	1
GH	0	0	0	1	0	0	1	1	0	0	0
BJ	0	0	0	1	1	1	0	1	1	1	0
ZK	0	0	0	0	0	1	1	0	1	0	0
DM	0	0	0	0	0	0	1	1	0	1	0
SB	0	0	0	0	1	0	1	0	1	0	1
NB	0	0	0	0	1	0	0	0	0	1	0

```
C:\Users\HiTech\Desktop\AI project\FinalCSP\FinalCSP.exe

Plz Enter The Neighbors Of Variable "BJ(=6)"(Zero Or one) :0 0 0 1 1 1 0 1 1 1 0

Plz Enter The Neighbors Of Variable "ZK(=7)"(Zero Or one) :0 0 0 0 0 0 1 1 0 1 0 0

Plz Enter The Neighbors Of Variable "DM(=8)"(Zero Or one) :0 0 0 0 0 0 0 1 1 0 1 0

Plz Enter The Neighbors Of Variable "SB(=9)"(Zero Or one) :0 0 0 0 1 0 1 0 1 0 1

Plz Enter The Neighbors Of Variable "NB(=10)"(Zero Or one) :0 0 0 0 1 0 0 0 0 1 0

The Neighbors Of Variable "TB"(=0)"     is :       BSH     FD      SR      KH
The Neighbors Of Variable "BSH"(=1)"    is :        TB     FD
The Neighbors Of Variable "FD"(=2)"     is :        TB    BSH      SR
The Neighbors Of Variable "SR"(=3)"     is :        TB     FD      KH      GH      BJ
The Neighbors Of Variable "KH"(=4)"     is :        TB     SR      BJ      SB      NB
The Neighbors Of Variable "GH"(=5)"     is :        SR     BJ      ZK
The Neighbors Of Variable "BJ"(=6)"     is :        SR     KH      GH      ZK      DM      SB
The Neighbors Of Variable "ZK"(=7)"     is :        GH     BJ      DM
The Neighbors Of Variable "DM"(=8)"     is :        BJ     ZK      SB
The Neighbors Of Variable "SB"(=9)"     is :        KH     BJ      DM      NB
The Neighbors Of Variable "NB"(=10)"    is :        KH     SB

**********************************************


The Solution is :

TB:Red
BSH:Green
FD:Blue
SR:Green
KH:Blue
GH:Blue
BJ:Red
ZK:Green
DM:Blue
SB:Green
NB:Red
--------------------------------
Process exited after 0.9665 seconds with return value 0
Press any key to continue . . .
```



*نقشه ی رنگ آمیزی شده به کمک خروجی بدست آمده

نمونه دوم :

استان های ایران:

تعداد رنگ ها : ۳

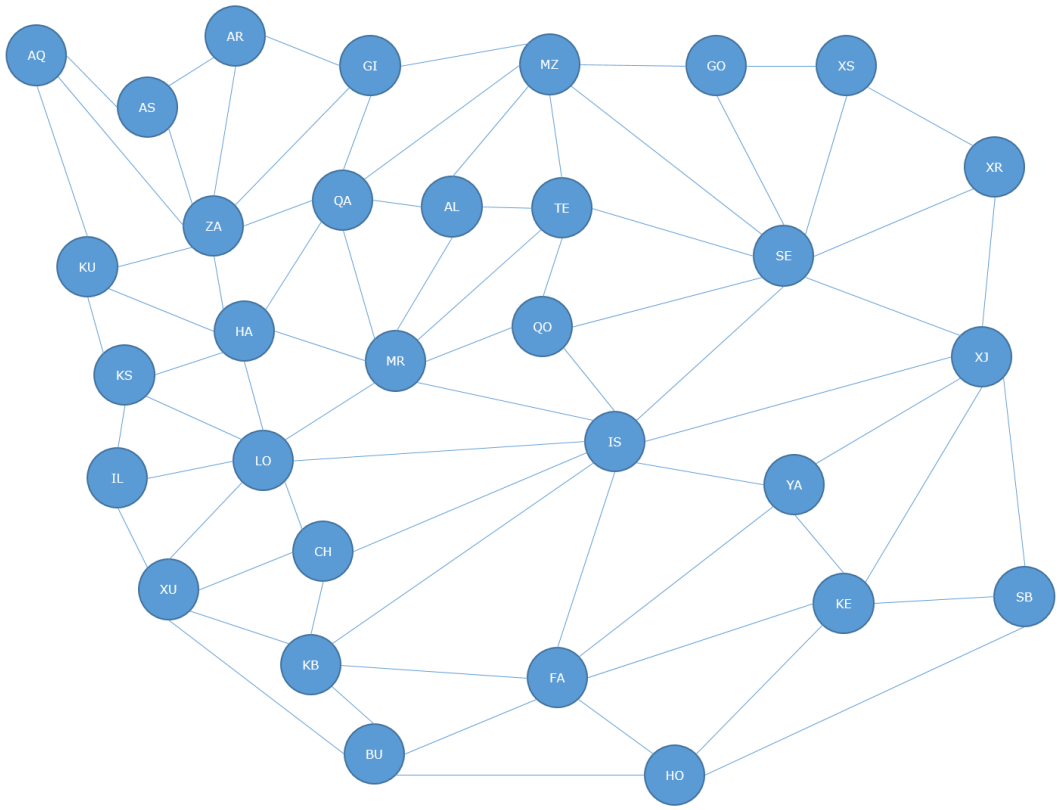
نام رنگ ها : Red,Green,Blue

تعداد متغیر ها: ۳۱

نام متغیر ها:

AL:	Alborz	**GI:**	Gilan	**KS:**	Kermanshah	**LO:**	Lorestan	**TE:**	Tehran
AR:	Ardabil	**GO:**	Golestan	**XS:**	Khorasan North	**MR:**	Markazi	**YA:**	Yazd
AS:	Azerbaijan East	**HA:**	Hamadan	**XR:**	Khorasan Razavi	**MZ:**	Mazandaran	**ZA:**	Zanjan
AQ:	Azerbaijan West	**HO:**	Hormozgan	**XJ:**	Khorasan South	**QA:**	Qazvin		
BU:	Bushehr	**IL:**	Iilam	**XU:**	Khuzestan	**QO:**	Qom		
CH:	Chahar Mahal and Bakhtiari	**IS:**	Isfahan	**KB:**	Kohgiluyeh and Boyer-Ahmad	**SE:**	Semnan		
FA:	Fars	**KE:**	Kerman	**KU:**	Kurdistan	**SB:**	Sistan and Baluchestan		

گراف:



Province	Abbreviation	Province	Abbreviation
Alborz	AL	Khorasan, Razavi	XR
Ardabil	AR	Khorasan, South	XJ
Azerbaijan, East	AS	Khuzestan	XU
Azerbaijan, West	AQ	Kohgiluyeh and Boyer-Ahmad	KB
Bushehr	BU	Kurdistan	KU
Chahar Mahaal and Bakhtiari	CH	Lorestan	LO
Fars	FA	Markazi	MR
Gilan	GI	Mazandaran	MZ
Golestan	GO	Qazvin	QA
Hamadan	HA	Qom	QO
Hormozgān	HO	Semnan	SE
Ilam	IL	Sistan and Baluchestan	SB
Isfahan	IS	Tehran	TE
Kerman	KE	Yazd	YA
Kermanshah	KS	Zanjan	ZA
Khorasan, North	XS		

ماتریس همجواری:

	AL	AR	AS	AQ	BU	CH	FA	GI	GO	HA	HO	IL	IS	KE	KS	XS	XR	XJ	XU	KB	KU	LO	MR	MZ	QA	QO	SE	SB	TE	YA	ZA
AL	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	1	0	0
AR	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
AS	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
AQ	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1
BU	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0
CH	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	1	0	1	0	0	0	0	0	0	0	0	0
FA	0	0	0	0	1	0	0	0	0	0	1	0	1	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1	0
GI	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	1
GO	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0
HA	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	1	1	0	1	0	0	0	0	0	1
HO	0	0	0	0	1	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
IL	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	0
IS	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	1	0	1	0	1	1	0	0	1	1	0	0	1	0
KE	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1	0	1	0
KS	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0
XS	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0
XR	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0
XJ	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	1	0	0	0	0	0	0	0	0	0	1	1	0	1	0
XU	0	0	0	0	1	1	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0
KB	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
KU	0	0	0	1	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
LO	0	0	0	0	0	1	0	0	0	1	0	1	1	0	1	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0
MR	1	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0	1	0	0	1	1	0	0	1	0	0
MZ	1	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	1	0	0
QA	1	0	0	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	1
QO	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	1	0	0
SE	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	1	1	1	0	0	0	0	0	1	0	1	0	0	1	0	0
SB	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
TE	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	1	1	0	0	0	0	0
YA	0	0	0	0	0	0	1	0	0	0	0	0	1	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
ZA	0	1	1	1	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0

```
C:\Users\HiTech\Desktop\AI project\FinalCSP\FinalCSP.exe

Plz Enter The Neighbors Of Variable "YA(=29)"(Zero Or one) :0 0 0 0 0 0 1 0 0 0 0 0 1 1 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0

Plz Enter The Neighbors Of Variable "ZA(=30)"(Zero Or one) :0 1 1 1 0 0 0 1 0 1 0 0 0 0 0 0 0 0 0 1 0 0 0 1 0 0 0 0 0 0

The Neighbors Of Variable "AL"(=0)"     is :        MR     MZ     QA     TE
The Neighbors Of Variable "AR"(=1)"     is :        AS     GI     ZA
The Neighbors Of Variable "AS"(=2)"     is :        AR     AQ     ZA
The Neighbors Of Variable "AQ"(=3)"     is :        AS     KU     ZA
The Neighbors Of Variable "BU"(=4)"     is :        FA     HO     XU     KB
The Neighbors Of Variable "CH"(=5)"     is :        IS     XU     KB     LO
The Neighbors Of Variable "FA"(=6)"     is :        BU     HO     IS     KE     KB     YA
The Neighbors Of Variable "GI"(=7)"     is :        AR     MZ     QA     ZA
The Neighbors Of Variable "GO"(=8)"     is :        XS     MZ     SE
The Neighbors Of Variable "HA"(=9)"     is :        KS     KU     LO     MR     QA     ZA
The Neighbors Of Variable "HO"(=10)"    is :        BU     FA     KE     SB
The Neighbors Of Variable "IL"(=11)"    is :        KS     XU     LO
The Neighbors Of Variable "IS"(=12)"    is :        CH     FA     XJ     KB     LO     MR     QO     SE     YA
The Neighbors Of Variable "KE"(=13)"    is :        FA     HO     XJ     SB     YA
The Neighbors Of Variable "KS"(=14)"    is :        HA     IL     KU     LO
The Neighbors Of Variable "XS"(=15)"    is :        GO     XR     SE
The Neighbors Of Variable "XR"(=16)"    is :        XS     XJ     SE
The Neighbors Of Variable "XJ"(=17)"    is :        IS     KE     XR     SE     SB     YA
The Neighbors Of Variable "XU"(=18)"    is :        BU     CH     IL     KB     LO
The Neighbors Of Variable "KB"(=19)"    is :        BU     CH     FA     IS     XU
The Neighbors Of Variable "KU"(=20)"    is :        AQ     HA     KS     ZA
The Neighbors Of Variable "LO"(=21)"    is :        CH     HA     IL     IS     KS     XU     MR
The Neighbors Of Variable "MR"(=22)"    is :        AL     HA     IS     LO     QA     QO     TE
The Neighbors Of Variable "MZ"(=23)"    is :        AL     GI     GO     QA     SE     TE
The Neighbors Of Variable "QA"(=24)"    is :        AL     GI     HA     MR     MZ     ZA
The Neighbors Of Variable "QO"(=25)"    is :        IS     MR     SE     TE
The Neighbors Of Variable "SE"(=26)"    is :        GO     IS     XS     XR     XJ     MZ     QO     TE
The Neighbors Of Variable "SB"(=27)"    is :        HO     KE     XJ
The Neighbors Of Variable "TE"(=28)"    is :        AL     MR     MZ     QO     SE
The Neighbors Of Variable "YA"(=29)"    is :        FA     IS     KE     XJ
The Neighbors Of Variable "ZA"(=30)"    is :        AR     AS     AQ     GI     HA     KU     QA

No Solution Found!

--------------------------------
Process exited after 1.039 seconds with return value 0
Press any key to continue . . .
```

❋ متاسفانه به ازای "سه رنگ" برنامه به پاسخی نرسید!

– سعی مجدد این بار با استفاده از ۴ رنگ : (Red,Green,Blue,Yellow)

```
C:\Users\HiTech\Desktop\AI project\FinalCSP\FinalCSP.exe
The Neighbors Of Variable "YA(=29)"     is :        FA     IS     KE     XJ
The Neighbors Of Variable "ZA(=30)"     is :        AR     AS     AQ     GI     HA     KU     QA

************************************************


The Solution is :

AL:Red
AR:Red
AS:Green
AQ:Red
BU:Red
CH:Green
FA:Green
GI:Green
GO:Red
HA:Red
HO:Blue
IL:Red
IS:Red
KE:Red
KS:Green
XS:Blue
XR:Red
XJ:Blue
XU:Yellow
KB:Blue
KU:Yellow
LO:Blue
MR:Green
MZ:Blue
QA:Yellow
QO:Blue
SE:Green
SB:Green
TE:Yellow
YA:Yellow
ZA:Blue
--------------------------------
Process exited after 11.05 seconds with return value 0
Press any key to continue . . .
```

*نقشه ی رنگ آمیزی شده به کمک‌خروجی بدست آمده

نمونه سوم :

ایالات استرالیا:

تعداد رنگ ها : ۳

نام رنگ ها : Red,Green,Blue

تعداد متغیر ها: ۷

نام متغیر ها: WA NT NSW Q V SA T

ماتریس همجواری:

	WA	NT	NSW	Q	V	SA	T
WA	0	1	0	0	0	1	0
NT	1	0	1	0	0	1	0
NSW	0	1	0	0	0	1	0
Q	0	0	1	0	1	1	0
V	0	0	0	1	0	1	0
SA	1	1	1	1	1	0	0
T	0	0	0	0	0	0	0