4.6.11 C# тілінің класс әдістері айнымалыларының локальді және глобальді түсінігі. Мысал.
4.6.12 C# тілінің класс әдістері денесінде қандай «аты» жазу операторы қолданылуы мүмкін?
4.6.13 Бір әдіс ішіне (мысалы, `voidMain()` басқа әдісті жариялауға бола ма? Мысал.
4.6.14 Рекурсия түсінігі. Мысал.
4.6.15 Рекурсия жетістігі және кемшілігі.

# 5 C# ТІЛІНІҢ КӨПӨЛШЕМДІ МАССИВІ

## 5.1 Бесінші жұмыс мақсаты

Көпөлшемді массивтерді жариялау ережелерін оқу, көпөлшемді матрицаларды қолданып, консоль қосымша құруға практикалық дағдылану.

## 5.2 Теориялық мәліметтер

5.2.1 Көпөлшемді массив түсінігі

Массивтерді бірөлшемді және көпөлшемді бөлу тарихи сипаттама болып табылады. Олардың арасында принципті айырмашылық жоқ. Бір өлшемді массивтер - көпөлшемді массивтердің арнайы жағдайы.

Бірөлшемді массивтер математикалық құрылымды көрсетуге мүмкіндік береді, оларға векторлар, екіөлшемді матрицалар, үшөлшемді - трехмерные – деректер текшелері, жоғары өлшемді массивтер – көпөлшемді деректер текшелері.

Массив өлшемі типтің сипаттамасы болып табылады. Көпөлшемді массивті мәлімдемені жазу форматы бойынша жариялау төмендегідей:

<тип>[, … ,] <объявители>;

Мысалы: int[,] matri; int[,,] kubi;

Массив өлшемі жақша ішінде үтір арқылы көрсетіледі.

Үтірлер саны 1-ге артық болса массив өлшемін көрсетеді. Бірөлшемді массивтердегі сияқты көпөлшемді массивтерде де жариялау болады. Біз оны граф тарауында қолданатын боламыз. Әдетте , көп өлшемді массивтер баптандыру массивін құру үшін бағдарламалық қамтамасыз жұмыс істейді.

5.2.2 Массивтердің массиві

C# тілінде массивтердің тағы бір түрі массивтердің массиві болып табылады, оларды басқаша «сынған» массивтер деп атайды (jaggedarrays).

Массивтердің массивін бірөлшемді массив деп қарастыруға болады, олардың элементі массив болып табылады, олардың элементінің элементі де массив болып табылады, белгілі бір деңгейге дейін жалғаса береді.

Қандай жағдайларда бұл деректер құрылымдарында қажет болуы мүмкін? Бұл массивтер ағаштарды ұсыну үшін пайдаланылуы мүмкін, онда түйіндер балалардың кез келген санын беруі мүмкін. Бұл, мысалы, отбасы ағаш болады.

Бірінші деңгей төбесі - Fathers, ікені білдіреді, бірөлшемді массив болып берілуі мүмкін, сондықтан Fathers[i] – бұл і-дің әкесі. Екінші деңгей төбесі массивтердің массиві - Children, сондықтан Children[i] – бұл массив і әкесінің баласы, ал Children[i][j] - і-ші әкенің j-ші баласы . Немерелерін көрсету үшін үшінші деңгей қажет, сондықтан GrandChildren [i][j][k] будет і-ші әкенің j-ші баласының к-ші немересін көрсетеді.

Массивтерді жариялаудың кейбір артықшылықтары бар. Егер көпөлшемді массивтердің өлшемдерін жариялағанда үтір қолданылса, «үзілген» массивтер үшін түсінірек белгі қолданылады – тік жақша жұптылығы; мысалы, int[][] массивті белгілейді, оның элементі - int типінің бірөлшемді массиві.

Ең қиыны массивтерді құру және баптау. Бұл жерде new int[3][5] конструкторын шақыруға болмайды, «үзілген» жиынтық массивін бермейді. Ең төменгі деңгейде әрбір массив үшін конструктор шақыру керек. Мұндай жариялау массивтердің күрделілігі болып табылады. Мысалы: masmasi бүтін типінің абстрактілі массивтің массивін жариялау төмендегідей:

```
int[][]masmasi = new int[3][]
  {
        newint[] {5,7,9,11},
        newint[] {2,8},
        newint[] {6,12,4}
};
```

Массив masmasi-дің тек екі деңгейі бар. Оның үш элементі бар деп санауға болады, оның әрбіреуі массив болып табылады. Ішкі массивті құру үшін әрбір массивке new конструкторын шақыру керек. Берілген мысалда ішкі массив элементтері мән иеленеді. Бұл массивті келесідей жариялауға және баптауға болады:

```
int[][]masmasi  = new int[3][]
{
        newint[4],
        newint[2],
        newint[3]
};
```

Бұл жағдайда массив элементтерін баптау кезінде нөлдік мән қабылдайды. Шынайы баптауды бағдарламалық жолмен орындау қажет. Айта кетерлік жағдай, жоғарғы деңгей конструкторында 3 тұрақтысын ескермей және жай new int[][] деп жазуға болады. Ең қызығы, бұл конструкторды шақырған кезде оны мүлдем ескермей қою:

```
int[][]masmasi  =
{
        newint[4],
        newint[2],
        newint[3]
};
```

Бірақ төменгі деңгей конструкторы қажет болып табылады.

5.2.3 Сызықтық алгебра алгоритмдері

Матрица дегеніміз m жолдан және n бағаннан тұратын сандар жиыны. Программист үшін матрица – екіөлшемді матрица. Егер m = n болса матрица квадраттық деп аталады, кері жағдайда тікбұрышты. m және n сандары матрица өлшемін анықтайды. Тікбұрышты матрицаны ауыстыру, қосу, көбейту операциялары анықтайды.

A - матрицасының өлшемі m*n болсын ( m жолдан және n бағаннан) $a_{i,j}$ элементімен. Ауыстыру матрицасы деп B = AT өлшемі n*m, оның элементі $b_{i,j} = a_{j,i}$ . Ауыстыру матрицасында берілген матрица жолы баған болып табылады.

$$A = \left\| \begin{array}{l} a_{1,1}, a_{1,2} \dots a_{1,n} \\ a_{2,1}, a_{2,2} \dots a_{2,n} \\ \dots \\ a_{m,1}, a_{m,2} \dots a_{m,n} \end{array} \right\| \quad B = A^T = \left\| \begin{array}{l} a_{1,1}, a_{2,1} \dots a_{m,1} \\ a_{1,2}, a_{2,2} \dots a_{m,2} \\ \dots \\ a_{1,n}, a_{2,n} \dots a_{m,n} \end{array} \right\|$$

Қосу операциясы бірдей өлшемді тікбұрышты матрицаны анықтайды. A, B, C - тікбұрышты матрицасы m*n өлшемді болсын. Онда матрицаның суммасы келесідей анықталады:

$$A = \left\| \begin{array}{l} a_{1,1}, a_{1,2} \dots a_{1,n} \\ a_{2,1}, a_{2,2} \dots a_{2,n} \\ \dots \\ a_{m,1}, a_{m,2} \dots a_{m,n} \end{array} \right\| \quad B = \left\| \begin{array}{l} b_{1,1}, b_{1,2} \dots b_{1,n} \\ b_{2,1}, b_{2,2} \dots b_{2,n} \\ \dots \\ b_{m,1}, b_{m,2} \dots b_{m,n} \end{array} \right\|$$

$$C = A + B = \left\| \begin{array}{l} a_{1,1} + b_{1,1}, a_{1,2} + b_{1,2} \dots a_{1,n} + b_{1,n} \\ a_{2,1} + b_{2,1}, a_{2,2} + b_{2,2} \dots a_{2,n} + b_{2,n} \\ \dots \\ a_{m,1} + b_{m,1}, a_{m,2} + b_{m,2} \dots a_{m,n} + b_{m,n} \end{array} \right\|$$

Көбейту матрицасы тікбұрышты матрицаларға анықталған, олардың бірінші көрсеткішінің бағандар саны екінші көрсеткіштің жолдар санына тең.

Матрица жұмысының жолдар саны бірінші көрсеткіштің жолдар санына, екінші көрсеткіштің бағандар санына тең.

A – матрицасының өлшемі m*p, B – өлшемі p*n, онда C матрицасы = A*B оның өлшемі m*n. Матрица жұмысының элементтері жүп элементтер ретінде анықталады, бірінші көрсеткіштің жолдар саны екінші көрсеткіштің бағандар санына сәйкес келеді.

$$A = ||a_{i,j}|| \quad i = 1, \dots m; \; j = 1, \dots p; \quad B = ||b_{j,k}|| \quad j = 1, \dots p; \; k = 1, \dots n$$

$$C = A * B = ||c_{i,k}|| \quad i = 1, \dots m; \; k = 1, \dots n;$$

$$c_{i,k} = \sum_{j=1}^{p} a_{i,j} * b_{j,k};$$

Көбейту ылғы тік және ауыстыру матрицасы үшін анықталған. Егер A – тікбұрышты матрицасының өлшемі m*n, онда ылғи B квадраттық матрица өлшемі m*m:

$$B = A * A^T = B^T = (A * A^T)^T = (A^T)^T * A^T = A * A^T$$

Мұндай жұмыстың нәтижесі симметриялық матрица. Квадраттық матрица симметрицлық деп аталады, егер $a_{i,j} = a_{j,i}$ барлық i және j, немесе егер $A = A^T$. Ауыстыру операциясы, қосу және көбейтудің келесідей қасиеттері бар:

$$(A^T)^T = A; \quad (A+B)^T = A^T + B^T; \quad (A*B)^T = B^T * A^T$$

5.2.4 Квадраттық матрицалар

Квадраттық матрица диагональндық деп аталады, егер барлық элементтер, диагональдық элементтерден басқа нөлге тең болса, яғни $a_{i,j} = 0$ егер $i/=j$.

Квадраттық матрица бірлік деп аталады, егер барлық элементтер, диагональдықтан басқа, нөлге тең болса, ал диагональдық элементтер бірге тең. Яғни $a_{i,j} = 0$ егер $i/=j$ және $a_{i,j} = 1$ егер $i = j$. Бірлік матрица Е әрпімен белгіленеді, және ол матрицаларды көбейткен кезде бірлік болып табылады:

$$A * E = E * A = A$$

Оның элементтерінің функциясын белгіленген квадраттық матрица үшін , айқындаушы болып табылады. Матрицаны анықтау сандар жиынтығы

$$D(A) = \begin{vmatrix} a_{1,1}, a_{1,2} \dots a_{1,n} \\ a_{2,1}, a_{2,2} \dots a_{2,n} \\ \dots \\ a_{n,1}, a_{n,2} \dots a_{n,n} \end{vmatrix}$$

айналасында бір желілері арқылы білдіреді :

Анықтауыш беретін функция маңызды қасиеттер атқарады.

Диагональдық матрица анықтауышы диагональдық элементтер мәніне тең. Бұдан, Е матрицасының анықтауышы 1 тең.

Матрицаның айқындаушы қарапайым өзгерістерді жүзеге асыру бойынша өзгерген жоқ. Қарапайым операциялар ( трансформациялау) астында басқа жолдар сызықтық комбинациясы матрицаның кез келген жолдың қосымша жатады. Атап айтқанда, егер j нөмерлі жолға $k(k/=j)$ нөмерлі жол қосса, кейбір санға көбейтілген, онда матрица анықтауышы өзгермейді.

Егер матрицаның барлық элементін кейбір q санына көбейтсе, онда матрица анықтауышы q ретке өзгереді (q көбейтіледі).

Егер j және k жолдардың орынын ауыстырса, онда анықтауыш модулі өзгермейді, бірақ белгі өзгереді, егер өзгешелік $|k-j|$ тақ сан болса.

Матрица анықтауышы анықтауыш мәніне тең:

$$D(A*B) = D(A) * D(B)$$

Ресми анықтама бермей, оның қасиеттері негізінде матрицаның анықтауышын есептеу алгоритмін қарастырайық.

Егер А квадраттық матрицасының анықтауышы нөлге тең болмаса, онда кері матрица бар, белгіленуі $A^{-1}$.Тікелей және кері матрицасы байланысты :

$$A * A^{-1} = A^{-1} * A = E$$

Төмендегідей матрицаны көшіру , көбейту және инверсия байланысты :

$$(A^T)^{-1} = (A^{-1})^T; \quad (A * B)^{-1} = B^{-1} * A^{-1}$$

Барлық анықтауышы бар бірөлшемді квадраттық матрицалар көбейту бойынша бір топ құрайды. Топта бірлік элемент бар, әрбір элемент үшін оған кері элемент болады.

Анықтауыштар n сызықтық теңдеулердегі n белгісі бар түбірді табу үшін кеңінен қолданылады.

### 5.3 Зертханалық жұмысты орындауға арналған мысал

0-ден 5-ке дейінгі диапазонда кездейсоқ толық санды A – 5*3 және B – 3*4 матрицасын тұрғызуды қарастырайық.

Бұл матрицаларды басып шығару керек, одан кейін A*B жұмысының нәтижесін ұсынатын C матрицасын тауып оны басу керек.

Әдістемелік нұсқаулықтың теориялық бөлімнің 5 модульінде матрицаларды көбейтудің алгоритмі қарастырылған. C матрицасының мәнін тексеру үшін A және B матрицасының мәндер диапазоны аз болып таңдалды.

Бағдарлама коды:

```
usingSystem;
using System.Collections.Generic;
using System.Linq;
using System.Text;
namespace ConsoleApplication1
{
classProgram
 {
publicstaticvoid sozd(int[,] ma, int[,] mb)
  {
Random rnd = newRandom();
Console.WriteLine("Матрицысозданы!!");
Console.WriteLine("МатрицаA 5*3:");
for (int i = 0; i < 5; i++)
   {
for (int j = 0; j < 3; j++)
    {
     ma[i, j] = rnd.Next() % 6;
Console.Write(ma[i, j] + "\t");
   }
Console.WriteLine();
   }
Console.WriteLine();
Console.WriteLine("МатрицаB 3*4:");
for (int i = 0; i < 3; i++)
  {
for (int j = 0; j < 4; j++)
    {
     mb[i, j] = rnd.Next() % 6;
Console.Write(mb[i, j] + "\t");
}
Console.WriteLine();
```

```
    }
Console.WriteLine();
  }

publicstaticvoid umnmatr(int[,] mc, int[,] ma, int[,] mb)
  {
int S;
for (int i = 0; i < 5; i++)
for (int j = 0; j < 4; j++)
     {
      S = 0;
for (int k = 0; k < 3; k++)
       S = S + ma[i, k] * mb[k, j];
      mc[i,j] = S;
}
Console.WriteLine("Матрица C 5*4:");
for (int i = 0; i < 5; i++)
   {
for (int j = 0; j < 4; j++)
    {
Console.Write(mc[i, j] + "\t");
    }
Console.WriteLine();
   }
Console.WriteLine();
  }

staticvoid Main()
  {
int[,] a = newint[5, 3];
int[,] b = newint[3, 4];
int[,] c = newint[5, 4];
int k = 0;
string buf;
while (k < 3)
   {
    Console.WriteLine("1 - Создать и напечатать матрицы A 5*3 и
B 3*4");
Console.WriteLine("2 - Найти и напечатать матрицу C = A * B");
Console.WriteLine("3 - Выход из программы");
Console.WriteLine("Введите пункт меню программы");
buf = Console.ReadLine();
    k = Convert.ToInt32(buf);
switch (k)
    {
case 1: sozd(a,b);  break;
case 2: umnmatr(c,a,b); break;
default: break;
    }
   }
  }
}
```

```
}
```

Бағдарлама жұмысы:

```
1 -   A 5*3 және B 3*4 матрицасын құрып және басып шығару
2 -   C = A * B матрицасын тауып оны басып шығару
3 – бағдарламадан шығу
Бағдарламаның мәзірін енгізіңіз
1
Матрицы құрылды!!
Матрица A 5*3:
2       1       3
5       5       3
2       0       2
4       4       0
4       2       3

Матрица B 3*4:
0       5       0       3
0       3       5       3
3       0       5       2

1 - A 5*3 және B 3*4 матрицасын құрып және басып шығару
2 - C = A * B матрицасын тауып оны басып шығару
3 - бағдарламадан шығу
Бағдарламаның мәзірін енгізіңіз
2
Матрица C 5*4:
9       13      20      15
9       40      40      36
6       10      10      10
0       32      20      24
9       26      25      24

1 - A 5*3 және B 3*4 матрицасын құрып және басып шығару

2 - C = A * B матрицасын тауып оны басып шығару
3 - бағдарламадан шығу
Бағдарламаның мәзірін енгізіңіз
```

### 5.4 Зертханалық жұмысқа арналған үй тапсырмасы

Минус 20-дан 60-қа дейінгі аралықтағы кездейсоқ бүтін сандардан тұратын 5×5 матрицасын құру керек. Матрицаны басып шығару керек. Максималды элементті тауып, ол орналасқан жол мен бағанды жою керек.

Қалған мәндерді жаңа 4×4 матрицасына көшіріңіз. Жаңа матрицаны басып шығарыңыз. Бағдарламада мәзір қолданыңыз.

## 5.5 СРС-қа арналған жеке тапсырма

5.5.1 Минус 50-ден 50-ге дейінгі аралықтағы кездейсоқ бүтін сандардан тұратын 5×5 матрицасын құру керек. Матрицаны басып шығару керек. Минималды элементті табу керек, егер ол бірінші жолда орналаспаса, онда ол ораналасқан жолды бірінші жолмен орын ауыстыру керек. Жаңа матрицаны басып шығару керек.
5.5.2 Мәтін дилог режимінде енгізіледі және сөздер «пробел» символымен ажыратылатын қарапайым сөйлемді құрайды. Жолдың ұзындығы 60 тең етіп және жол бойында «пробел» символдары біртекті болатындай етіп қосымша «пробел» символдарды қосу керек.
5.5.3 0-ден 100-ге дейінгі аралықтағы кездейсоқ бүтін сандардан тұратын A6×6 матрицасын құру керек. Матрицаны басып шығару керек.Әр жолда орналасқан сандарды кему ретімен орналастыру керек. Жаңа матрицаны басып шығару керек.
5.5.4 Жолда сөздері «пробел» символымен ажыратылатын қарапайым сөйлем бар. Максималды және минималды ұзындықты сөздерді басып шығару керек.
5.5.5 Минус 20-ден 70-ге дейінгі аралықтағы кездейсоқ бүтін сандардан тұратын A5×5 матрицасын құру керек. Матрицаны басып шығару керек. Максималды элементті тауып, ол орналасқан бағанды жою керек. Қалған мәндерді B5×4 матрицасына көшіру керек. Жаңа матрицаны басып шығару керек.
5.5.6 Диалог режимінде енгізілген жолда екі жақшадан аспайтын C# тілінің шарты if операторы бар. Диалог режимінде енгізілген шарт дұрыс жазылғанын тексеру керек.
5.5.7 Минус 30-ден 50-ге дейінгі аралықтағы кездейсоқ бүтін сандардан тұратын A6×6 матрицасын құру керек. Матрицаны басып шығару керек. Матрицаның басынқы және бағыңынқы диагональда орналасқан сандарды өсу ретімен орналастыру керек. Жаңа матрицаны басып шығару керек.
5.5.8 Диалог режимінде енгізілген жол қарапайым сөйлемді құрайды және сөздер «пробел» символымен ажыратылады. Максималды және минималды ұзындықты сөздерді тауып, олардың орындарын ауыстырыңыз, жаңа жолды экран мониторына шығарыңыз.
5.5.9 Минус 50-ден 50-ге дейінгі аралықтағы кездейсоқ бүтін сандардан тұратын A6×6 матрицасын құру керек. Матрицаны басып шығару керек. Оң және теріс сандырдың санын табыңыз. Егер модуль бойынша теріс сандардың соммасы оң сандардыңсоммасынан көп болса, онда оң сандардың соммасы артық болмағанша теріс сандардың минус таңбасын плюс таңбасына ауыстыру керек. Жаңа матрицаны басып шығару керек.

5.5.10 Диалог режимінде енгізілген жол қарапайым сөйлемді құрайды және сөздер «пробел» символымен ажыратылады. Жолдың сөздерін әріптердің өсу ретімен орналастырып, жаңа жолды қалыптастырыңыз.
5.5.11 0-ден 9-ға дейінгі аралықтағы кездейсоқ бүтін сандардан тұратын A10×10 матрицасын құру керек. Матрицаны басып шығару керек. Матрица қанша 0, 1, 2, …, 9 сандардың бар екенің есептеңіз.
5.5.12 Диалог режимінде енгізілген жол қарапайым сөйлемді құрайды және сөздер «пробел» символымен ажыратылады. A K×2 матрицаны құрыңыз, онда барлық K сөздерінің бастапқы және соңғы позициялардың нөмердерін енгізіңіз.
5.5.13 0-ден 20-ға дейінгі аралықтағы кездейсоқ бүтін сандардан тұратын A6×6 матрицасын құру керек. Матрицаны басып шығару керек. Матрицаның бағандарын элементтердің соммаларының өсу ретімен орналастыру керек. Жаңа матрицаны басып шығару керек.
5.5.14 0-ден 20-ға дейінгі аралықтағы кездейсоқ бүтін сандардан тұратын A6×6 матрицасын құру керек. Матрицаны басып шығару керек. Матрицада қайталанатын сандарынан тұратын векторды құрыңыз және басып шығарыңыз.
5.5.15 Диалог режимінде енгізілген жол қарапайым сөйлемді құрайды және сөздер «пробел» символымен ажыратылады. Мәтінде қайталанатын сөздерін анықтаңыз. Қайталанатын сөздерді және сөйлемде қайталау жиілігі бойынша экран мониторына шығарыңыз.
5.5.16 Минус 20-дан 60-қа дейінгі аралықтағы кездейсоқ бүтін сандардан тұратын A7×7 матрицасын құру керек. Матрицаны басып шығару керек. Матрицаның басты және бағаңынқы диагональдарынан «төмен» жатқан элементтердің соммасын табу керек.
5.5.17 0-ден 20-ға дейінгі аралықтағы кездейсоқ бүтін сандардан тұратын A5×5 матрицасын құру керек. Матрицаны басып шығару керек. Матрицада қайталанатын сандарды және олардың қайталану жиілігін табыңыз және басып шығарыңыз.
5.5.18 Диалог режимінде енгізілген жол қарапайым сөйлемді құрайды және сөздер «пробел» символымен ажыратылады. Сөйлемде ең жиі кездесетін символды тауып, экран мониторына шығарыңыз.
5.5.19 Минус 50-ден 50-ге дейінгі аралықтағы кездейсоқ бүтін сандардан тұратын A8×8 матрицасын құру керек. Матрицаны басып шығару керек. Басты диагональға паралель барлық диагональдарды басып шығарыңыз.
5.5.20 0-ден 20-ға дейінгі аралықтағы кездейсоқ бүтін сандардан тұратын A10×10 матрицасын құру керек. Матрицаны басып шығару керек. Ең көп бірдей сандар бар жолды тауып, оның нөмерін экран мониторына шығарыңыз.

## 5.6 СРС-да есеп-хатты қорғауға арналған бақылау сұрақтары

5.6.1 Көпөлшемді массив түсінігі. Мысалдар.
5.6.2 Көпөлшемді массив қалай жарияланады? Мысал.

5.6.3 0-ден 10-ға дейінгі диапазонда A5×5 матрицасын қалай тұрғызу керек мысал келтір.
5.6.4 Экран мониторына A5×5 матрицасын қалай енгізуге болады мысал келтір.
5.6.5 A5×5 матрицасындағы үлкен санды іздеу алгоритмін түсіндіріңіз.
5.6.6 Массивтердің массив түсінігі. Сондай массивтерді жариялау. Мысал.
5.6.7 Екі матрицаны қосу алгоритмні түсіндіріңіз. Мысал.
5.6.8 Екі матрицаны көбейту алгоритмні түсіндіріңіз. Мысал.
5.6.9 C# тілінің жол айнымалы түсінігі. Мысал.
5.6.10 C# тілінде escape-қолданбасы не үшін керек? Мысалдар.
5.6.11 C# тілінде жолдық айнымалылар қалай салыстырылады? Мысал.
5.6.12 Қалай кейбір белгілі фрагментті мәтіннен жоюға болады? Мысал.
5.6.13 Кейбір белгілі фрагменті мәтінге қалай қоюға болады? Мысал.
5.6.14 Қалай мәтіндегі таңбалар санын анықтауға болады? Мысал.
5.6.15 Split және Join әдістерін тағайындау. Мысалдар.

# 6 ГРАФТЫҢ ӨТУ АЛГОРИТМІ

## 6.1 Алтыншы тақырыптың мақсаты

Граф теориясының негізгі түсініктерін оқу және граф алгоритмінің орындалуын қарау, «транспорттық» есепті шешуге арналған консоль қосымшасында орындауға практикалық дағдылану.

## 6.2 теориялық мәлімет

6.2.1 Қатаңдату ағаштары. Негізгі түсініктер

Графтар теориясында ағаш түсінігінің қарапайым екі анықтамасы бар. Теоремаларды қорытпай, дұрыстығын дәлелдеу үшін анықтамасын келтірейік. Бірінші анықтама, байланысқан граф ағаш болып табылады, онда доға саны төбе санынан бірге кіші.

Екінші анықтама анықтырақ – ағаш циклдар жоқ қосылған граф болып табылады. Ағаш , кейбір қабырғаны алып тастау арқылы граф алу, осы ағаштың графы немесе жақтауы деп аталады.

Мысал, 6.1 суретте «жұлдыз» граф көрсетілген бірнеше созылмалы ағашпен берілген. 6.1 суретте созылмалы ағаштың тағы бір түрі көрсетілген.