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.

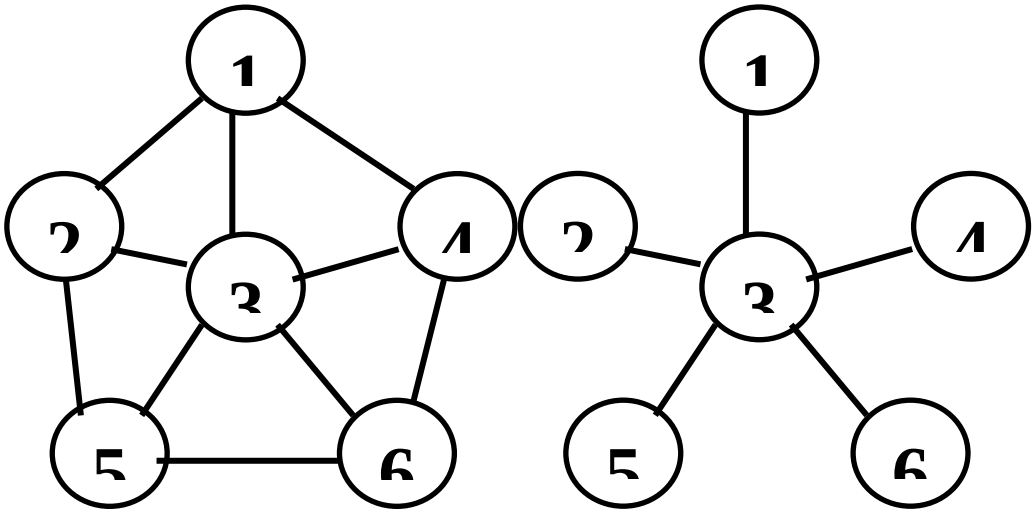
6.2 теориялық мәлімет

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.



Сурет 6.1 – Байланысқан граф және оның созылмалы ағашы.

Граф теориясында қолданылмаған қабырға (созылған ағаш жойылған граф қабырғасы арқасында алынды) хорда деп аталады.

Егер созылған ағашқа еркін хорданы қосса, алынған граф бір циклды болады.

Көп циклді граф арқасында алгоритм құрылады, қолданбалы сипаттамасы бар, мысалы, Кирхгоф заңына сәйкес электронды сымдарға анализ жүргізу.

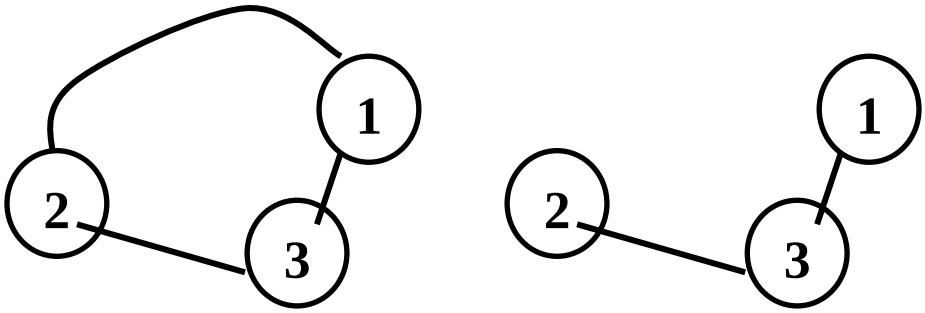
6.2.2 Созылған ағашты тұрғызу алгоритмі

Созылған ағаштың табудың әртүрлі алгоритмі бар.

Созылған ағашты табудың алгоритмін қарастырайық, онда бәріне белгілі Дейкстр алгоритмі қолданылады – графтың берілген төбесінен қалған төбелеріне дейінгі ең аз маршрут.

Бұл мүмкін, өйткені соны дәлелдейтін теория бар, берілген граф төбесінен қалған төбелеріне дейінгі ең аз қашықтық арқылы созылған ағаш тұрғызуға болады.

Граф төбелері арасындағы ең аз қашықтықты табудың алгоритмін қарастырайық. Мысалы, іргелес үш төбе бар 1,2 және 3. Төбелер арасындағы қашықтық 1 - 2 тең 5, төбелер 2 – 3 тең 3, төбелер 3 – 1 тең 1. Маршрутты таңдау 1 – 2 граф төбелері үшін 1 – 3 – 2 маршрут қолданыады . Қабырға 1 – 2 графтан өшіріледі.

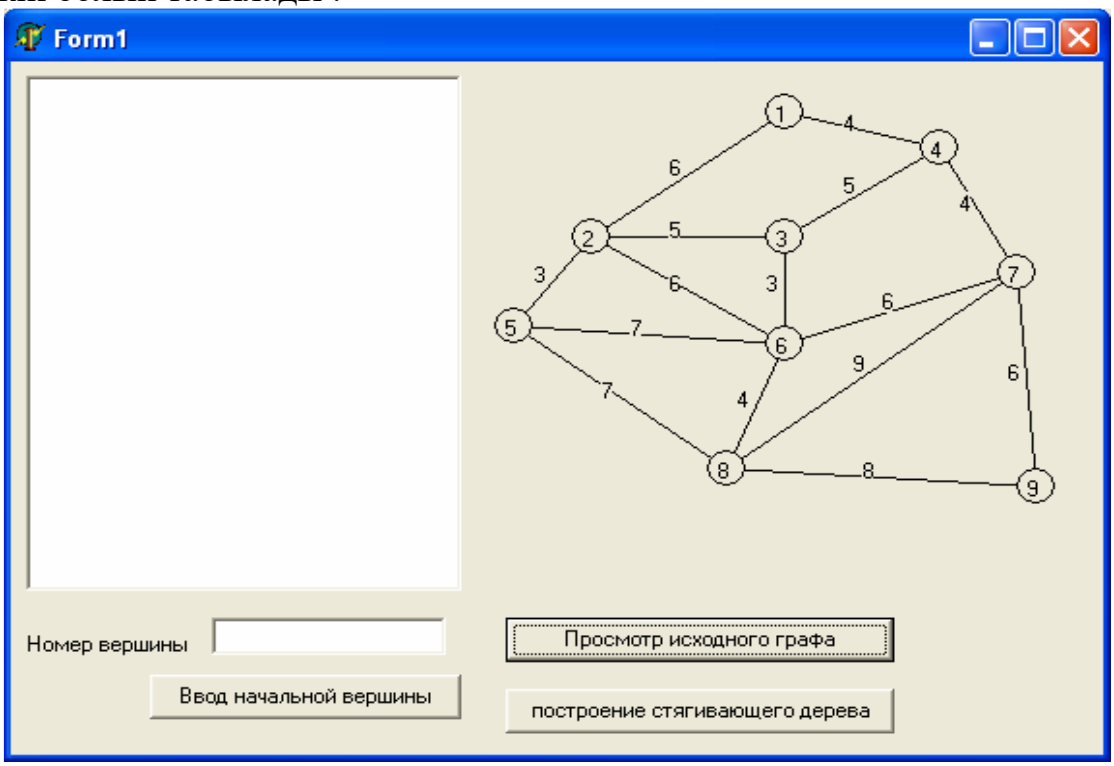


Сурет 6.2 – ең аз маршрутты табу алгоритмі

Көрсетілген алгоритмді қолдану созылған ағашты тұрғызуға көмек береді. «көпмөлшер» граф төбесі алгоритімде төбелердің өсу ретімен орындалады (for циклі қолданылған) , онда екі маршрут «тең» болғанда созылған ағаш болып бірінші нұсқа қалады.

Есепті бағдарламалау үшін бастапқы графты байланысқан граф ретінде созылған ағашты тұрғызу Delphi бағдарламалау ортасында бар (6.3 суретте граф көрсетілген).

Созылған ағашты құру алгоритмі Дейкстрдің ең аз қашықтықты табу алгоритмін қолданады. Осы алгоритмнің бұрын талқыланған бағдарламаны жүзеге асырудан айырмашылығы ол бастапқы төбелердің мәнін орнату үшін мүмкін болып табылады .



Сурет 6.3 – Бастапқы байланысқан граф

6.2.3 Дейкстр алгоритмі

Дейкстр алгоритмін үш массивті қолдануды талап етеді, олардың өлшемі граф төбелеріне сәйкес келеді.

Бірінші массив, «тұрақты» төбе массиві, берілген граф төбесінен қалған төбелерге дейінгі ең аз қашықтықты сақтайды. Алдымен бұл массивке бастапқы төбе нөмері жазылады (төбелер, басқа граф төбелеріне дейінгі ең аз қашықтықты табу үшін). Массив атауы Дейкстр алгоритіміндегі таңдап алынған төбелер атауымен сәйкес келеді, ең алдымен барлық граф төбелері «уақытша» болып жарияланады, ал таңдап алынған төбелер «тұрақты» болып табылады.

Екінші массив барлық басқа төбелерге ең аз арақашықтықты сақтау үшін пайдаланылады. Алдымен төбенің таңдап алған нөмеріне сәйкес іргелес матрица жолы жазылады.

Логикалық типті үшінші массивте таңдап алынған (тұрақты) граф төбелері көрсетіледі. Бастапқыда «тұрақты» граф төбесі алғашқы төбе болып табылады.

Әрі қарай «уақытша» төбе таңдап алынады, онда бастапқы (тұрақты) төбеге дейінгі қашықтық ең аз. Бұл төбе k төбесі болып табылады. Міндетті түрде «тұрақты» төбеден «уақытша» төбеге дейін барлық қашықтық тексеріледі, сол сияқты k төбесінен де тексеріледі. Ең аз қашықтық екінші массивте жазылады, ал бірінші массивке k жазылады, егер ең аз қашықтық k төбесі арқылы өтсе.

Одан кейін k төбесі «тұрақты» болып жарияланады (бұл үшінші массивте жазылады) және жаңа «тұрақты» төбеге сәйкес графтың келесі төбесін іздеу алгоритмі жалғасады.

6.3 Зертханалық жұмысты орындау мысалы

6.1 есеп. 6.3 суретте көрсетілген байланысқан графқа сәйкес созылған ағаш тұрғызу. Графтың алғашқы төбе нөмерін диалог режімінде беру. Созылған ағаш үшін бастапқы төбеден барлық қалған төбелеріне дейін маршруттың тізімін көрсету.

Бағдарлама коды:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace ConsoleApplication1
{
classProgram
 {
publicstaticint[,] a = newint[9, 9]
   {{   0,   6,1000,   4,1000,1000,1000,1000,1000},
    {   6,   0,   5,1000,   3,   6,1000,1000,1000},
    {1000,   5,   0,   5,1000,   3,1000,1000,1000},
    {   4,1000,   5,   0,1000,1000,   4,1000,1000},
    {1000,   3,1000,1000,   0,   7,1000,   7,1000},
```

```
    {1000,   6,   3,1000,   7,   0,   6,   4,1000},
    {1000,1000,1000,   4,1000,   6,   0,   9,   6},
    {1000,1000,1000,1000,   7,   4,   9,   0,   8},
    {1000,1000,1000,1000,1000,1000,   6,   8,   0}};

publicstaticint[] d = newint[10];
publicstaticint[] post = newint[10];
publicstaticbool[] t = newbool[10];
publicstaticint i, j, p, k, minras, begver;

publicstaticvoid poisk()
{
string buf;
//Ввод значения переменной a в режиме диалога
Console.WriteLine("Введите номер вершины графа целое значение 0
- 8");
buf = Console.ReadLine();
   begver = Convert.ToInt32(buf);
//начальные установк
for (i = 0; i < 9; i++)
{
     post[i] = begver;
     t[i] = true;
d[i] = a[begver,i];
    }
   t[begver] = false;
   post[begver] = 0;
   minras = 0;
for (i = 0; i < 8; i++)
    {
// поисквершины k
     minras = 1000;
for (j = 0; j < 9; j++)
if ((t[j] == true) && (minras > d[j]))
{
        minras = d[j]; k = j;
}
// поиск маршрутов и минимальных расстояний через вершину k
t[k] = false;
for (j = 0; j < 9; j++)
if ((t[j] == true) && (d[j] > d[k] + a[k,j]))
      {
       d[j] = d[k] + a[k,j];
       post[j] = k;
      }
    }
  }


publicstaticvoid print()
{
Console.WriteLine("Печатаемматрицусмежности : ");
```

```
for (i = 0; i < 9; i++)
    {
for (j = 0; j < 9; j++)
Console.Write("\t" + a[i, j]);
Console.WriteLine();
    }
Console.WriteLine();
// печатьномероввершинграфа
for (i = 0; i < 9; i++) Console.Write("\t" + i);
Console.WriteLine();
// печать массива минимальных расстояний
for (i = 0; i < 9; i++) Console.Write("\t" + d[i]);
Console.WriteLine();
// печатьмассивамаршрутов
for (i = 0; i < 9; i++) Console.Write("\t" + post[i]);
Console.WriteLine();
Console.WriteLine();
int dlin;
// печать маршрутов стягивоющего дерева
for (i = 0; i < 9; i++)
    {
     dlin = d[i];
Console.Write("Путь № {0}-{1} длина пути = {2}\t\t{3}", i,
begver, dlin,i);
p = i;
if (i != begver)
      {
do
        {
         p = post[p];
if (p!=0) Console.Write("\t" + p);
}
while (p != 0);
Console.WriteLine();
      }
elseConsole.WriteLine();
    }
  }

staticvoid Main()
  {
   poisk();
   print();
Console.ReadLine();
  }
 }
}
```

Бағдарлама жұмысы:

0 - 8 бүтін мәннің граф төбелері санын енгізіңіз

4

Іргелес матрицаларды басып шығарамыз :

0 6 1000 4 1000 1000 1000 1000 1000

```
   6     0     5  1000     3     6  1000  1000  1000
1000     5     0     5  1000     3  1000  1000  1000
   4  1000     5     0  1000  1000     4  1000  1000
1000     3  1000  1000     0     7  1000     7  1000
1000     6     3  1000     7     0     6     4  1000
1000  1000  1000     4  1000     6     0     9     6
1000  1000  1000  1000     7     4     9     0     8
1000  1000  1000  1000  1000  1000     6     8     0

0   1   2   3   4   5   6   7   8
9   3   8  13   0   7  13   7  15
1   4   1   2   0   4   5   4   7

Жол  № 0-4 жол ұзындығы = 9          0   1   4
Жол № 1-4 жол ұзындығы = 3           1   4
Жол № 2- жол ұзындығы = 8            2   1   4
Жол № 3-4 жол ұзындығы = 13          3   2   1   4
Жол № 4-4 жол ұзындығы = 0           4
Жол № 5-4 жол ұзындығы = 7           5   4
Жол № 6-4 жол ұзындығы = 13          6   5   4
Жол № 7-4 жол ұзындығы = 7           7   4
Жол № 8-4 жол ұзындығы = 15          8   7   4
```

6.4 Зертханалық жұмысқа үй тапсырмасы

Графты «тереңдігі» бойынша жүріп өту алгоритмін жүзеге асыратын алгоритмді бағдарлама құрыңыз. Өз графын қолданыңыз. Төбелер саны 10 кем болмау керек.

6.5 СРС арналған жеке тапсырма

6.5.1 Граф жасау - әуе тасымалы түйіндерді одан әрі түртіндінің өрісті қамтитын - облыс орталықтары атауы. Елдің барлық басқа орталықтарына берілген облыс орталығынан бастап, ең төменгі туристік маршрут іздеу қамтамасыз ету.

6.5.2 Егер граф жасау 10 шыңдары , белсенді элементтер кейбір электрондық тізбек сәйкес түйіндерінің кем емес . диалог режимінде орнатылған тізбектің тораптары арасындағы әрбір схеманың қарсылығын табыңыз.

6.5.3 граф түрі құрылымын енгізу схемасы қалалық автобус маршруттары . Түйіндері автобус маршруттарының құрылымына сәйкес келеді және одан әрі аялдама атын қамтиды . аялдама атына байланысты бағдарларды қамтамасыз ету .

6.5.4 Студенттер тобының атауларының графын жасау (атауы бірдей қолдануға рұқсат) . диалог режимінде көрсетілген аты бойынша студенттерге арналған іздеуді қамтамасыз ету.
6.5.5 Отбасылық ағашы 12 төбемен көрсетілген болып табылады. Әрбір төбесінде құрастыру одан әрі түрлі мүшесі кіреді. «тереңдігі» бойынша жүріп өтіу графы көмегімен барлық әйел затын табуды ұйымдастыру.
6.5.6 Қалалық төрт трамвай маршруты граф типінің құрылымы боцынша көрсетілген. Түйіндері трамвай маршруттарынан құрылымына сәйкес келеді және одан әрі аялдамасына атын қамтиды. Диалог режимінде енгізілген екі аялдамалар атауларының үшін , (аялдама саяхат қашықтыққа ең төменгі сомасы бойынша) екінші аялдамасына бірінші аз көші-қон маршрут табу үшін
6.5.7 Отбасылық ағашы 14 төбемен көрсетілген болып табылады. Әрбір төбесінде құрастыру одан әрі түрлі мүшесі кіреді. Ең жиі ерлер мен әйелдер атауларының іздеуді ұйымдастыру.
6.5.8 Қалалық төрт трамвай маршруты граф типінің құрылымы боцынша көрсетілген. Түйіндері трамвай маршруттарынан құрылымына сәйкес келеді және одан әрі аялдамасына атын қамтиды. Диалог режимінде енгізілген екі аялдамалар атауларының үшін , (аялдама саяхат қашықтыққа ең төменгі сомасы бойынша) екінші аялдамасына бірінші аз көші-қон маршрут табу үшін.
6.5.9 5 Отбасылық ағашы 15 төбемен көрсетілген болып табылады. Әрбір шыңында құрастыру одан әрі тектес және осы мамандық бойынша оның жұмысының уақыт негізгі қызметін қамтиды . Максималды жүгіру уақыт іздеу мамандық ұйымдастыру .
6.5.10 Қалалық төрт трамвай маршруты граф типінің құрылымы боцынша көрсетілген. Түйіндері трамвай маршруттарынан құрылымына сәйкес келеді және одан әрі аялдамасына атын қамтиды. Бұл тоқтау арқылы бағыттар санының кему тәртібімен қала аялдамалары атауын теріңіз.
6.5.11 Оның тораптары қалта каталогына сәйкес және қосымша қалта атын қамтиды граф ұсынылған компьютерлік диск - түрі құрамынан иерархиялық каталогтар құрылымы. Дискідегі бірдей қалта бар-жоғын анықтау үшін . Оларды басып шығару және оларға түбірлік каталогына жолын көрсетеді.
6.5.12 Егер граф жасау оның тораптары одан әрі түртіндінің өрісті қамтиды 10 төбедег кем емес. (Қосымша сипаты өрісі « дыбыстардың » символы бар) шыңдары ғана «мөлдір» « терең » бойынша графтар айналып алгоритмі әзірлеу .
6.5.13. Оның тораптары қалта каталогына сәйкес және қосымша қалта атын қамтиды граф ұсынылған компьютерлік диск - түрі құрамынан иерархиялық каталогтар құрылымы. Диск ойындары деп аталатын қалтаны тауып қанша рет анықтау үшін .
6.5.14 Егер граф жасау оның тораптары одан әрі түртіндінің өрісті қамтиды 10 төбеден кем емес . (Қосымша сипаты өрісі « дайын » символы бар) шыңдары «сәйкес » тек « терең » бойынша графтар айналып алгоритмі әзірлеу .
6.5.15 Диаграмма түрін (10 шыңы кем емес) құрылымын ұсынуға автобус маршруттары аудандық ауданы схемасы . Түйіндері ауылы ауданы атауы

құрылымына сәйкес келеді. Аялдама атына сәйкес маршруттар номерін көруді қарастыру. Ауылының атауы үшін , (аялдама саяхат қашықтыққа ең төменгі сомасы бойынша) осы ауылға аудан орталығынан аз көші-қон маршрут табу үшін, диалог енген .

6.5.16 Желінің өткізу қабілетін (секундына берілетін судың көлемі) бөлігі - қала , аудан Сумен жабдықтау желісі бағытталған кем дегенде 15 төбе графы, және доға ұсынылған. Су максималды сомасы түйін X. A , B және C тораптар жеткізілетін болуы мүмкін анықтаңыз. Барлық тораптар мәні диалог режимінде орнатылады. (Бұл қорғасын - доғаның қамтамасыз ете алады , егер) қабылдайтын тораптары арасындағы жем су біркелкі бөлу пайдаланыңыз.

6.5.17 Егер граф жасау 10 төбеден , белсенді элементтер кейбір электрондық тізбек сәйкес түйіндерінің кем емес. Диалог режимінде (параллель және сериялық қосылу схемасын ескеріңіз) белгіленген түйіндер схемасын арасындағы тізбектің кедергісі табыңыз.

6.5.18 Лабиринт 16 төбеден кем емес графпен көрсетілген, төбелері қмылыстарға сәйкес келеді. Лабиринттің кіру және шығу түйіндері белгілі. лабиринт жүруінің ең аз қашықтығын табу.

6.5.19 Қалалық төрт трамвай маршруты граф типінің құрылымы боцынша көрсетілген. Түйіндері трамвай маршруттарынан құрылымына сәйкес келеді және одан әрі аялдамасына атын қамтиды. Екі аялдамалар атауларының үшін , екінші аялдамасы (жасалған трансплантация саны , бірақ аялдамалары саны) бірінші аз көші-қон маршрут табу үшін , диалог режимінде енгізілген. Жинақтарын немесе өзгерту қажет аялдамалары және трамвай маршруттары бөлмелер атауларының жолын басып шығару, отыруға немесе өзгерту үшін сол қажеттілігі .

6.5.20 Қалалық төрт трамвай маршруты граф типінің құрылымы боцынша көрсетілген. Түйіндері трамвай маршруттарынан құрылымына сәйкес келеді және одан әрі аялдамасына атын қамтиды. Екі аялдамалар атауларының үшін , екінші аялдамасы (жасалған трансплантация саны , бірақ аялдамалары саны) бірінші аз көші-қон маршрут табу үшін , диалог режимінде енгізілген. Жинақтарын немесе өзгерту қажет аялдамалары және трамвай маршруттары бөлмелер атауларының жолын басып шығару, отыруға немесе өзгерту үшін сол қажеттілігі .

6.6 Контрольные вопросы для защиты отчета на СРСП

6.6.1 Граф түсінігі. Мысалдар.

6.6.2 Сырқаттанушылық түсінігі. Мысал.

6.6.3 Жалғастыру түсінігі. Мысал.

6.6.4 Граф жолы түсінігі. Графтың қандай жолы қарапайым деп аталады? Мысал.

6.6.5 Қандай граф байланысқан деп аталады? Мысал.

6.6.6 ЭВМ жадысында граф туралы мәліметті қалай сақтауға болады?

6.6.7 ЭВМ жадысында графты іргелес төбе тізімі ретінде қалай түсіндіруге болады?
6.6.8 Граф төбелері арасындағы ең аз қашықтықты іздеу алгоритмі.
6.6.9 Граф төбелері арасындағы ең аз қашықтықты іздеу алгоритмі.
6.6.10 «тереңдігі» бойынша графтың өту алгоритмі.
6.6.11 «ені» бойынша графтың өту алгоритмі.
6.6.12 «созылған» ағаш алгоритмін тұрғызу.
6.6.13 Екі төбе арасындағы барлық маршрутты табу алгоритмі.А
6.6.14 Неге «қайтару» жоғарғы жаңа меншік төбелері барлық циклдердің іздеу алгоритмі ?
6.6.15 Іздеу алгоритмі диаграммада екі берілген төбелер арасындағы барлық бағыттарды пайдаланылану құрылымын түсіндіріңіз .

7 Ұсынылған әдебиеттер тізімі

7.1. Негізгі әдебиет

7.1.1 Презентации лекций по дисциплине «Алгоритмизация и основы программирования» для студентов специальности 5В070400 – смотри портал кафедры ИС http : \\ www.do.ektu.kz
7.1.2 В.В. Фаронов Создание приложений с помощью C# Руководство программиста. - М.: “Эксмо”, 2008г.
7.1.3Т.А. Павловская C#, Программирование на языке высокого уровня. Учебник для вузов, СПб.: Питер, 2009г.
7.1.4Д. Кнут. Искусство программирования для ЭВМ. Т.3./ Сортировка и поиск / - М.:Мир,1976.

7.2 Қосымша әдебиет

7.2.1 Э. Йодан Структурное программирование и конструирование программ. М.: ”Мир”, 1989г.
7.2.2 Н. ВиртАлгоритмы и структуры данных. М. Изд-во «МИР», 1989г.
7.2.3 Э.ТроелсенC# и платформа .NET Библиотека программиста, СПб,: Питер, 2007г.