ՀԱՅԱՍՏԱՆԻ ՀԱՆՐԱՊԵՏՈՒԹՅԱՆ

ԿՐԹՈՒԹՅԱՆ, ԳԻՏՈՒԹՅԱՆ, ՄՇԱԿՈՒՅԹԻ և ՍՊՈՐՏԻ ՆԱԽԱՐԱՐՈՒԹՅՈՒՆ

ՀԱՅԱՍՏԱՆԻ ԱԶԳԱՅԻՆ ՊՈԼԻՏԵԽՆԻԿԱԿԱՆ ՀԱՄԱԼՍԱՐԱՆԻ

ՍԻՆՈՓՍԻՍ ԿՐԹԱԿԱՆ ԴԵՊԱՐՏԱՄԵՆՏԻ

ԻՆՏԵԳՐԱԼ ՍԽԵՄԱՆԵՐԻ և ՀԱՄԱԿԱՐԳԵՐԻ ՄՇԱԿՄԱՆ ԾՐԱԳՐԱՅԻՆ ՄԻՋՈՑՆԵՐԻ ԱՄԲԻՈՆ





ԿՈՒՐՍԱՅԻՆ ԱՇԽԱՏԱՆՔ

**ԽՈՒՄԲ** \*\*\*\*\*\*\*

**ԱՌԱՐԿԱ** \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**ԹԵՄԱԱ** \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**ԴԱՍԱԽՈՍ** \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**ՈՒՍԱՆՈՂ** \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**ՊԱՇՏՊԱՆՈՒԹՅՈՒՆԸ** \*\*\*\*\*\*\*\*\*\*

2022

Բովանդակություն

Ներածություն……………………………………………………………………………………3

Խնդրի պահանջը……………․․․․․…………………………………………………………….5

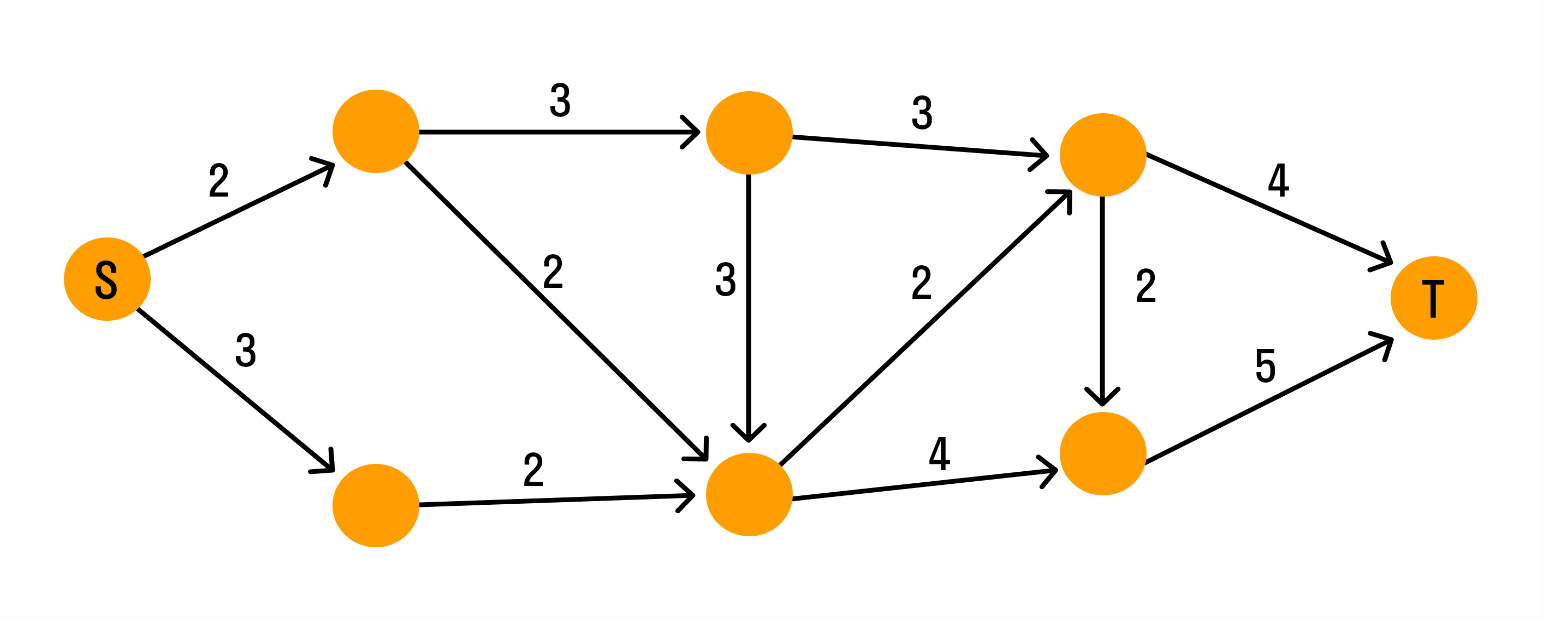
Լուծում……………………………………………………………………………………………..6

Ալգորիթմ․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․7

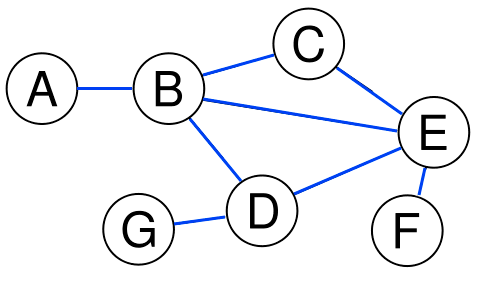
Վիզուալիզացիա․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․․9

**Ներածություն**

**Ինչ է գրաֆը?** Դիցուք –ը ցանկացած ոչ դատարկ վերջավոր բազմություն է, և –ն V բազմության տարրերի բոլոր ոչ կարգավոր զույգերի բազմությունն է, այնպես որ : Ենթադրենք, որ *E* ⊆ : կարգավոր զույգին կանվանենք գրաֆ, և այն կնշանակենք *G*-ով:

**** *G* = գրաֆը բաղկացած է գագաթներից(vertices)` V, և գագաթներն իրար միացնող կողերից(edges)` E: Գրաֆները լինում են մի քանի տեսակների՝ պարզ, ոչ պարզ, կապակցված, ոչ կապակցված, ուղղորդված, ոչ ուղղորդված և այլն։ Տվյալ կուրսային աշխատանքում մեզ հետաքրքիր են պարզ, ոչ ուղղորդված և կապակցված գրաֆները։

**Ինչ է պարզ գրաֆը?**

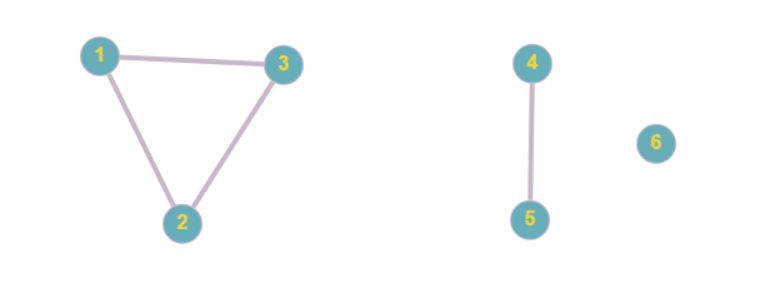
**** Հանգույց կանվանենք այն կողը, որը դուրս է գալիս v գագաթից և մտնում նույն v գագաթի մեջ, այլ կերպ ասած հանգույց կոչվող կողի կից գագաթները նույնն են։

Եթե (v1, v2) հարևան գագաթները ունեն 1-ից ավել ընդհանուր կող, ապա ասում են, որ տրված գագաթները ունեն բազմակողմ կողեր։ Եթե գրաֆը չունի հանգույցթներ և բազմակողերը, ապա այդպիսի գրաֆը կոչվում է պարզ գրաֆ։

**Կապակցված գրաֆ**

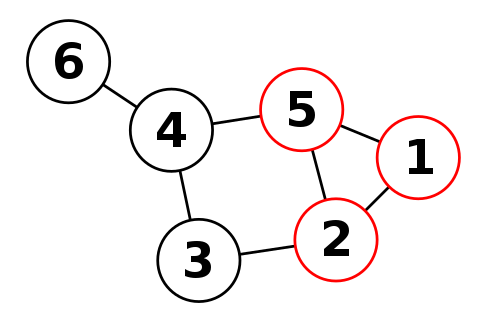
Նախ սահմանենք գրաֆների մեջ ուղի հասկացությունը։ Գրաֆի ուղին(կամ շղթան) գագաթների վերջավոր հաջորդականությունն է, որում յուրաքանչյուր գագաթ (բացի վերջինից) կողով միացված է հաջորդ գագաթի հետ։

Կապակցված գրաֆը այն գրաֆն է, որը պարունակում է ուղիղ մեկ կապակցվածության բաղադրիչ: Սա նշանակում է, որ այս գրաֆի ցանկացած (v1, v2) գագաթների միջև կա առնվազն մեկ ուղի:

Կապակցված(1, 2, 3) և ոչ կապակցված(4, 5, 6) գրաֆների օրինակ՝

**Ինչ է ցիկլը?**

Ցիկլը կանվանենք այն ուղին, որտեղ առաջին և վերջին գագաթները համընկնում են: Այս դեպքում ցիկլի երկարությունը նրա կազմող կողերի թիվն է։ Ընդ որում, եթե u և v գագաթները ինչ-որ կողի կից են, ապա այս սահմանման համաձայն (u,v,u) հաջորդականությունը ցիկլ է։



Նկարում պատկերված գրաֆը ունի 3 ցիկլ՝  
1 => 2 => 5

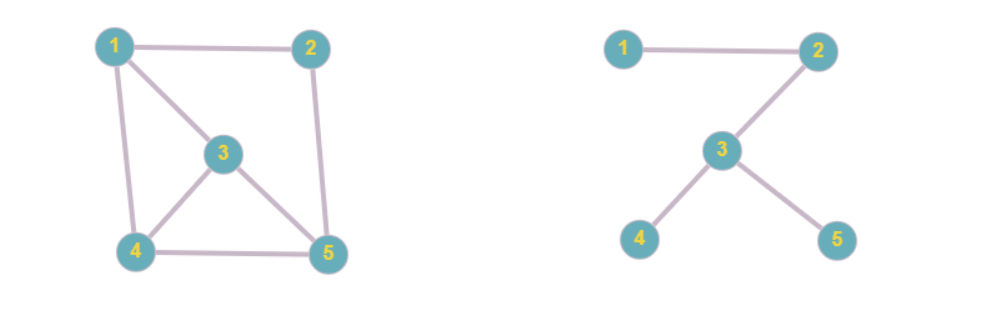
1 => 2 => 3 => 4 => 5

2 => 3 => 4 => 5

**Կմախքային ծառ**

Կմախքային ծառ է կոչվում տրված *G* = գրաֆից ստացված այն կապակցված ենթագրաֆը, որը չի պարունակում ցիկլեր, և պարունակում է սկզբնական գրաֆի բոլոր գագաթները։ Կմախքային ծառ ստանալու համար պարտադիր պայման է, որ սկզբնական գրաֆը լինի կապակցված։

Տրված գրաֆի և նրա կմախքային ծառի օրինակ՝



**Խնդրի պահանջը**

Տրված G(V, E) վերջավոր կապակցված գրաֆի կմախքային ծառի որոնման ալգորիթմի մշակում և ծրագրային իրացում:

**Կատարված աշխատանքը**

Github: <https://github.com/dpetrosy/graph_course_project>

**Լուծումը**

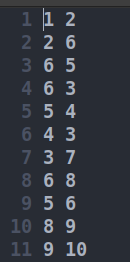
Ընդհանուր ծրագիրը կազմված է Makfile-ից, և data, include, src թղթապանակներից։ data թղթապանակում պահվում են ծրագրին անհրաժեշտ տվյալները, include-ում .hpp ֆայլերը, իսկ src-ում ծրագրի կոդերն են(.cpp): Makefile-ը ունի մի շարք թարգեթներ՝ all, clean, fclean, gcov, lcov և այլն։

Ծրագիրը աշխատացնելու համար անհրաժեշտ է clone անել git-ի րեպոզիտորիան։ Այնուհետև պետք է քաշել graphviz ծրագիրը վիզուալիզացիայի համար(եթե չկա համակարգչի վրա), linux-ի համար հավաքեք հետևյալ հրամանը՝ **sudo apt-get install graphviz:**

Graphviz-ը հաջողությամբ տեղադրոլուց հետո մեր ծրագիրը պատրաստ է օգտագործման, հավաքեք make re, և սպասեք .exe(executable) ֆայլի գեներացիային։ Գրաֆի տվյալները գտնվում են data/input.txt ֆայլում, իսկ ծրագրից ստացված տեղեկությունները մեզ են ներկայացվում արդեն վիզուալիզացված՝ նկարի տեսքով։

**Ֆայլից գրաֆի տվյալների ստացում**

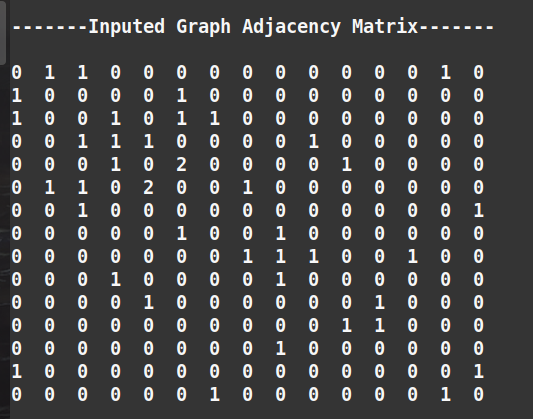
Ինչպես նշեցի, ծրագիրը գրաֆի տվյալները ստանում է data/input.txt ֆայլից, որտեղ տվյալները ունեն հետևյալ տեսքը՝

**** Ֆայլում ամեն տող իրենից ներկայացնում է գրաֆի կող, նրա աջ և ձախ կից գագաթներով։ Կարդալով ֆայլը ծրագիրը տվայլերը հավաքում է վեկտորի մեջ, որը պահում է կողերի ստրուկտուրաներ՝ edge, որն իր հերթին ունի համապատասխան աջ և ձախ գագաթների դաշտեր։ Կան նաև վալիդացիաներ, որոնք վերլուծուն են օգտագործողի կողմից ներմուծված գրաֆը, ու անհասկանալի կամ ոչ վալիդ մուտքագրված տվյալներից ծրագիրը կփորձի ստեղծել գրաֆ, եթե չստացվի՝ կտեղեկացնի օգտագործողին համապատասխան

նամակով։ Ներմուծումը գործում է մինչև 5-անիշ թվերի համար, բայց կարող է հեշտությամբ փոխվել ծրագրի մեջ։ Հաջորդ քայլը հարևանության մատրիցայի ստեղծում ու կապակցվածության ստուգումն է։

**Հարեվանության մատրից**

Գրաֆը ծրագիր մեջ պահվում է հարևանության մատրիցի միջոցով՝

Այն իրենից ներկայացնում է դինամիկ հիշողությունում տեղակայված երկչափ զանգված, որի ամեն էլեմենտի ինդեքսները ներկայանում են որպես գագաթներ, իսկ էլեմենտի արժեքը՝ տվյալ գագաթները միացնող կողերի քանակն է։ Օրինակ՝ matrix[5][8] = 2 նշանակում է, որ 5 և 8 գագաթները իրար հետ միացված են 2 կողով։

**Գրաֆի պարզեցում** Քանի որ մենք պետք է ստանանք կմախքային ծառ, ապա պետք է պարզեցնել գրաֆը՝ ազատվելով բազմակողմերից և հանգույցներից։

Սա կատարվում է հարեվանության մատրիցի միջոցով՝ անցնելով մատրիցով և 1 հավասարեցնելով այն էլեմենտները, որոնց արժեքը մեծ է 1-ից։ Նաև պետք է զրոյացնենք գլխավոր անկյունագծի էլեմենտները, հենց նրանք են պատասխանատու հնգույցների համար։

**Ալգորիթմ**

Կապակցվածության և կմախքային ծառի որոնումը իրագործված է DFS - Depth for Search ալգորիթմի միջոցով։ Ալգորիթմի հիմքում ռեկուրսիան է։ Այն շրջանցում է գրաֆը՝ մտնելով յուրաքանչյուր գագած ընդամենը 1 անգամ։ Ամեն անգամ ռեկուրսիվ կանչելով ալգորիթմի ֆունկցիան դիտարկում ենք տվյալ գագաթին հարևան գագաթները և ընտրում համարով ամենափոքրը՝ որն դեռ չենք այցելել ալգորիթմի միջոցով։ Ամեն այցելած գագաթ գունավորում ենք (պահելով տվյալ գագաթի համարի տակ true արժեք), որն էլ հաստատում է, որ ցանկացած գագաթ ալգորիթմի աշխատանքի ժամանակ կայցելենք 1 անգամ։ Երբ որ տվյալ գագաթը չունի այցելության գագաթներ՝ նրա բոլոր հարևանները գունավորված են, ալգորիթմը կանգ է առնում։ Ալգորիթմի բարդությունը Օ(V), որտեղ V գագաթների քանակն է, իսկ վատագույն դեպքում Θ(V+E).

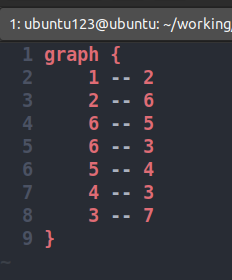
Տվյալ ալգորիթմի միջոցով հեշտությամբ կարող ենք ստուգել գրաֆի կապակցվածությունը։ Եթե ալգորիթմի աշխատանքից հետո մնացել են չգունավորված գոնե 1 գագաթ, ուրեմն գրաֆը կապակցված չէ։ Եթե օգտագործողի մուտքագրած գրաֆը կապակցված չլինի, նա կստանա համապատասխան հաղորդագրություն և ծրագիրը կխնդիր օգտագործողին վերանայել տվյալները։

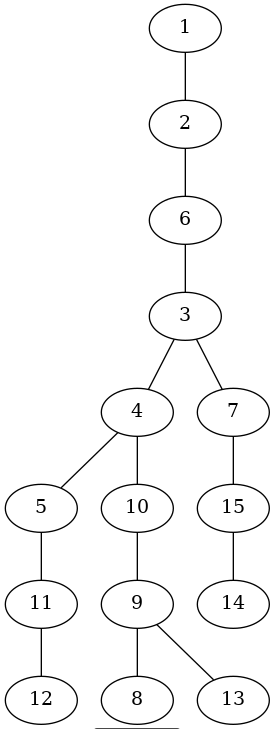
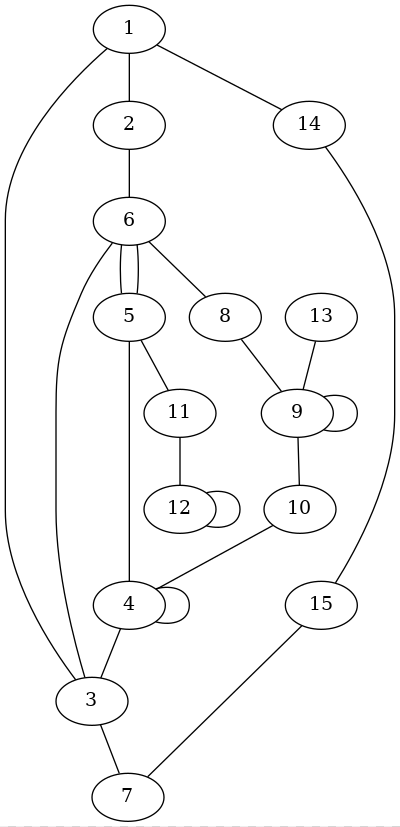
**** Կապակցվածությունը ստուգելուց հետո սկսվում է կմախքային ծառի որոնման պրոցեսը նույնցպես dfs-ով։ Ոնց նշել էի, կմախքային ծառը տրված գրաֆի ենթագրաֆ է, որը չի պարունակում ցիկլեր, և կապակցված է։ Քանի որ dfs ալգորիթմը ամեն գագաթ մտնում է 1 անգամ, իսկ ցիկլը այն ուղին է, որտեղ առաջին և վերջին գագաթները նույնն են, հետևաբար կարող ենք վստահորեն ասել, որ dfs-ի աշխատանքից հետո կստանանք մեր ուզած կմախքային ծառը։

**Վիզուալիզացիա**

Վիզուալիզացիան կատարվել է graphviz-ի միջոցով՝ Graphviz-ը բաց կոդով գրաֆիկների վիզուալիզացիայի ծրագիր է: Գրաֆի վիզուալիզացիան կառուցվածքային տեղեկատվությունը որպես վերացական գրաֆների և ցանցերի դիագրամներ ներկայացնելու միջոց է: Այն ունի կարևոր կիրառություններ ցանցերի, կենսաինֆորմատիկայի, ծրագրային ապահովման ճարտարագիտության, տվյալների բազայի և վեբ ձևավորման, մեքենայական ուսուցման և այլ տեխնիկական տիրույթների տեսողական ինտերֆեյսներում: Այն գրված է python-ով և օգտագործում է DOT լեզուն։

Graphviz-ից օգտվելու համար պետք է նախորոք տեղադրել այն համակարգչի վրա։ Պետք է ստեղծել .dot ֆայլ և գրել այնտեղ անհրաժեշտ կոդը(ինչպես նշվեց dot-ը լեզու է), որպեսզի graphviz-ը կարդա ֆայլը և վիզուալիզացնի մեր տված գրաֆը։ .dot ֆայլերը ունեն հետևյալ տեսքը՝

Որտեղ graph {} ը բանալի բառ է, իսկ 1 -- 2․․․ տալիս ենք կողերը, որոնցով պիտի կարուցվի գրաֆը։ Ծրագիրը օգտագործողի գրաֆը ստանալուց հետո կսարքի input.dot ֆայլ, իսկ ծառը գտնելուց հետո կսարքի tree.dot ֆայլ, որոնք հենց կօգտագործվենք graphviz-ի կողմից։ Ծրագրի վերջում ստանալու ենք 2 նկար՝ առաջինը օգտագործողի մուտքագրած գրաֆի նկարը, 2-րդը տրված գրաֆում գտնված կմախքային ծառի նկարը։

**Օգտագործողի մուտքագրած գրաֆի և գտնված ծառի պատկերների օրինակ՝**

**Գրականության ցանկ**

«Գրաֆների տեսություն» Պ․Ա․ Պետրոսյան, Վ․ Վ․ Մկրտչյան, Ռ․Ռ․ Քամալյան

«Դիսկրետ մաթեմատիկայի դասընթաց» Ռ․ Ն․ Տոնոյան

DFS <https://www.programiz.com/dsa/graph-dfs>

<https://sites.google.com/site/pythontutorarticles/obhod-v-glubinu>

Graphviz <http://graphviz.org/>

Գրաֆների դասընթաց <https://stepik.org/course/126/syllabus>