

Algoritmusok és adatszerkeztek II.

5. gyakorlat

Tartalom:
Mélységi keresés

- ❖ Mélységi keresés
- ❖ Irányított gráfok éleinek osztályozása
- ❖ Lejátszás, példa
- ❖ Lejátszás, gyakorló feladat
- ❖ Lejátszás irányítatlan gráfra
- ❖ Implementálási kérdések
- ❖ Fejtörő kérdések

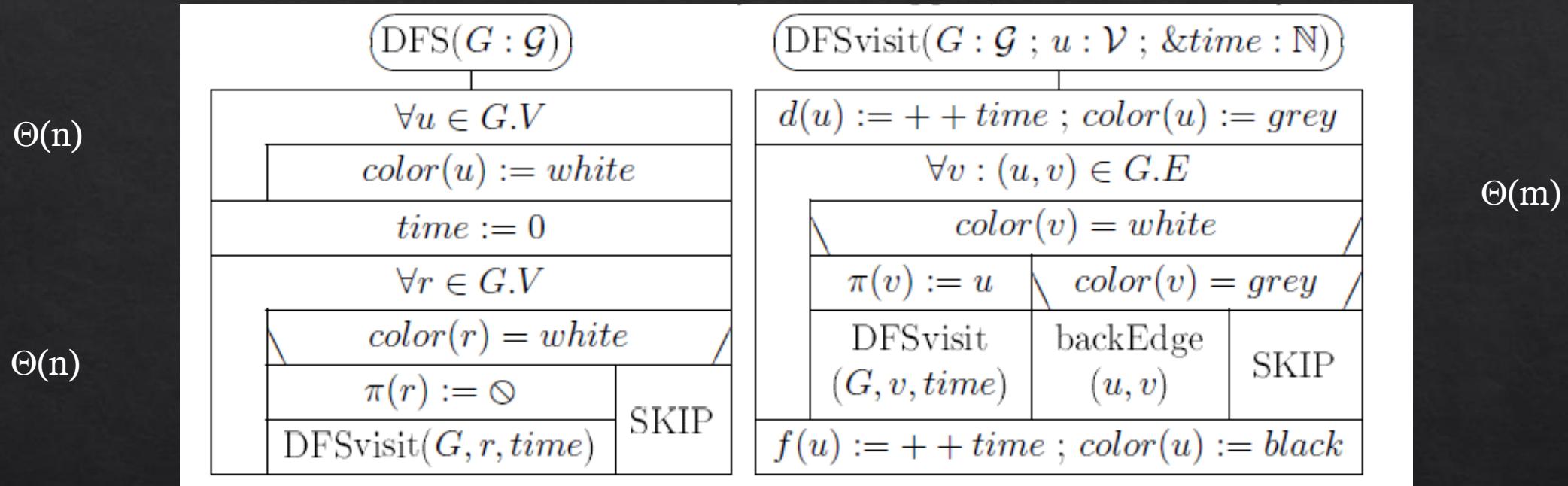
Mélységi keresés (mélységi bejárás)

- ❖ „Cseréljük ki a szélességi bejárásban a sort egy veremre, és megkapjuk a mélységi bejárást.”
- ❖ Bejáró algoritmusként használható irányítatlan/irányított gráfokon, főbb alkalmazási területe azonban az irányított gráfokkal kapcsolatos feladatok esetén van.
- ❖ Eredmények:
 - ❖ Mélységi erdő - szülő csúcsok megadása a csúcsoknál : $\pi(u)$
 - ❖ Idő „bélyegző”: megadja csúcsok felfedezésnek időpontját: $d(u)$ - discovery time és a feldolgozás befejezésének időpontját: $f(u)$ – finishing time
 - ❖ Színezést is használ: fehér/szürke/fekete, jelentése hasonló, mint a szélességi bejárásnál

Néhány nevezetes alkalmazási lehetőség

- ❖ Irányított gráf éleinek osztályozása, irányított kör megtalálása
- ❖ Körmentes irányított gráfok csúcsainak topologikus rendezése
- ❖ Irányított gráfok erősen összefüggő komponenseinek meghatározása
- ❖ Irányított gráfok félíg összefüggőségének eldöntése

Az algoritmus, és a műveletigénye



$$G = (V, E)$$

$$|V| = n$$

$$|E| = m$$

$$mT(n, m) = MT(n, m) \in \Theta(n + m)$$

Élek osztályozása irányított gráfban

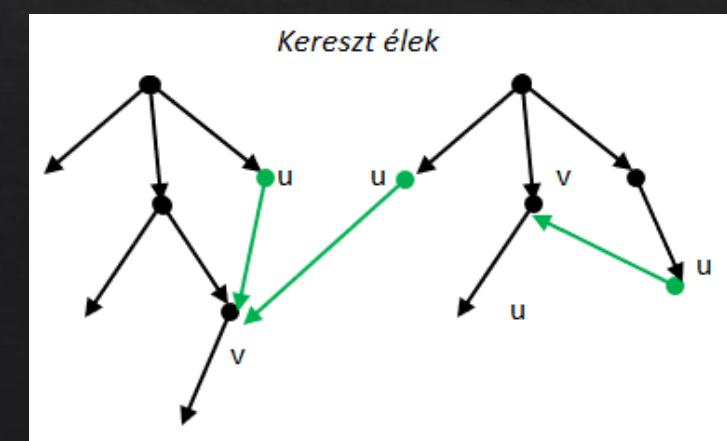
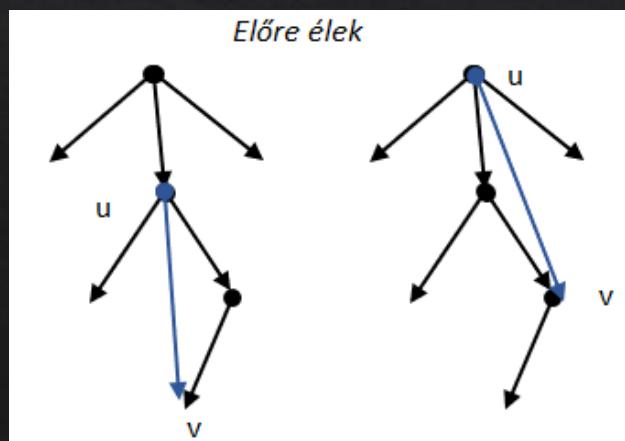
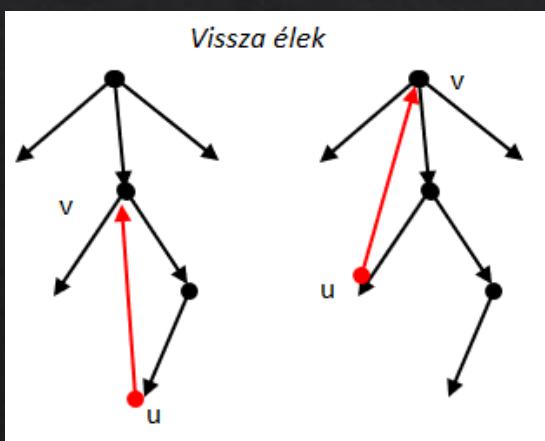
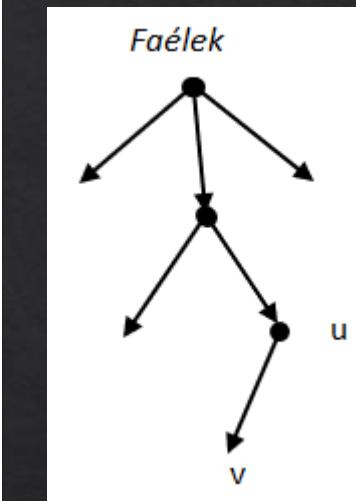
3.7. Definíció. A gráf éleinek osztályozása:

(u, v) fa-él (tree edge) $\iff (u, v)$ valamelyi mélységi fa egyik éle.
 (A fa-élek mentén járjuk be a gráfot.)

(u, v) vissza-él (back edge) $\iff v$ az u œse egy mélységi fában.

(u, v) előre-él (forward edge) $\iff (u, v)$ nem faél, de v az u leszármazottja egy mélységi fában.

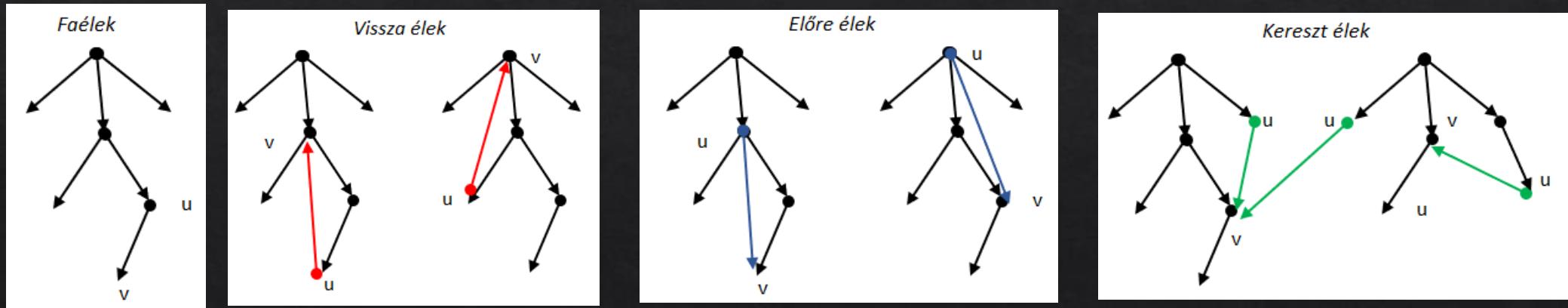
(u, v) kereszt-él (cross edge) $\iff u$ és v két olyan csúcs, amelyek ugyanannak a mélységi fának két különböző ágán vannak, vagy két különböző mélységi fában találhatók.



Élek osztályozása az algoritmusban

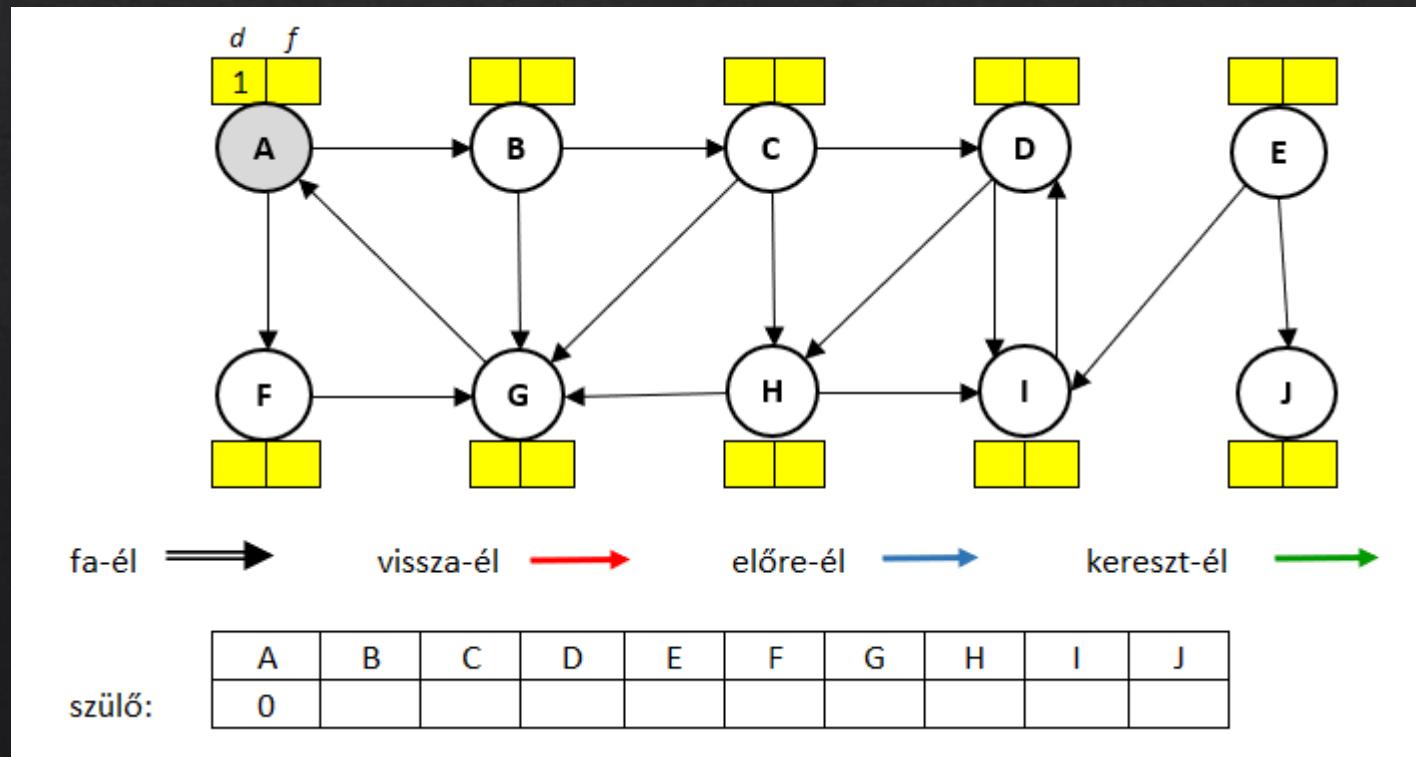
- ❖ Fa-él: $\text{color}(v)=\text{white}$
- ❖ Visszaél: $\text{color}(v)=\text{grey}$
- ❖ Előre-él: $\text{color}(v)=\text{black}$ és $d(u) < d(v)$
- ❖ Kereszt-él: $\text{color}(v)=\text{black}$ és $d(u) > d(v)$

$\text{DFS}(G : \mathcal{G})$	$\text{DFSvisit}(G : \mathcal{G} ; u : \mathcal{V} ; \&time : \mathbb{N})$
$\forall u \in G.V$	$d(u) := ++time ; \text{color}(u) := \text{grey}$
$\text{color}(u) := \text{white}$	$\forall v : (u, v) \in G.E$
$time := 0$	$\text{color}(v) = \text{white}$
$\forall r \in G.V$	$\pi(v) := u$
$\text{color}(r) = \text{white}$	$\text{color}(v) = \text{grey}$
$\pi(r) := \otimes$	$\text{DFSvisit}(G, v, time)$
$\text{DFSvisit}(G, r, time)$	$\text{backEdge}(u, v)$
	SKIP
	$f(u) := ++time ; \text{color}(u) := \text{black}$



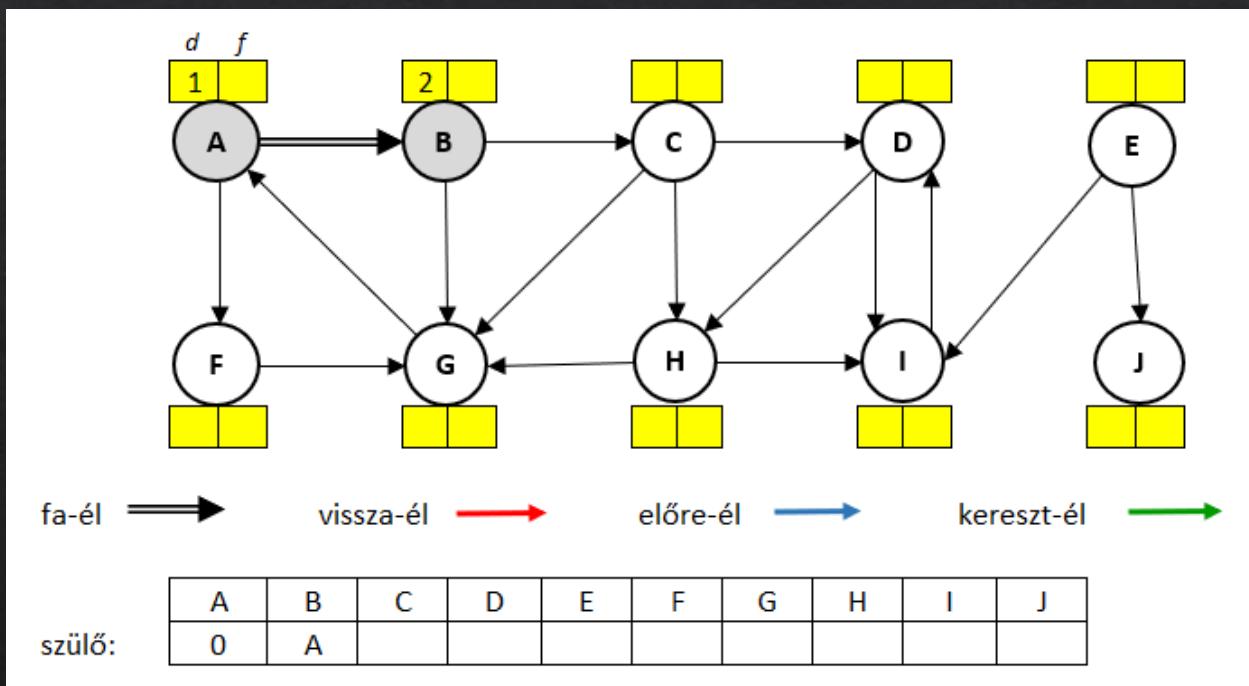
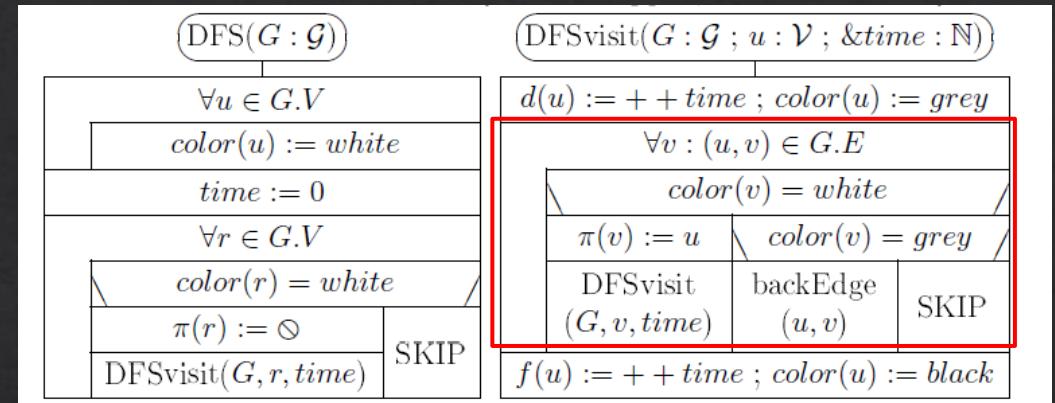
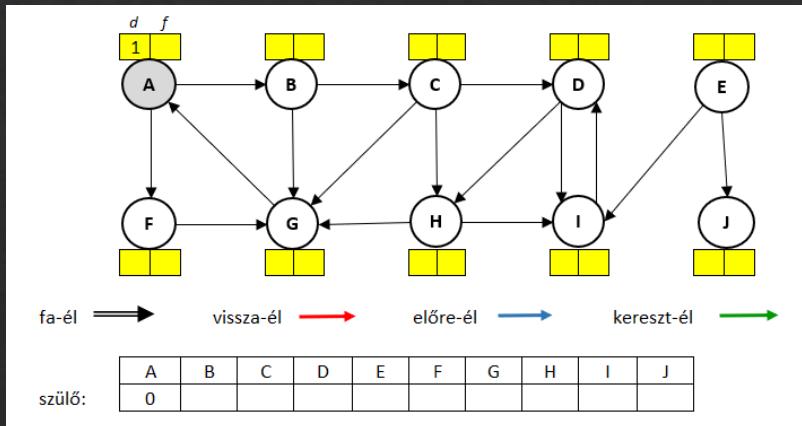
Mélységi bejárás lejátszása, az élek osztályozásával

Csúcsok, szomszédok sorba vétele ábécé szerinti sorrendben történik!

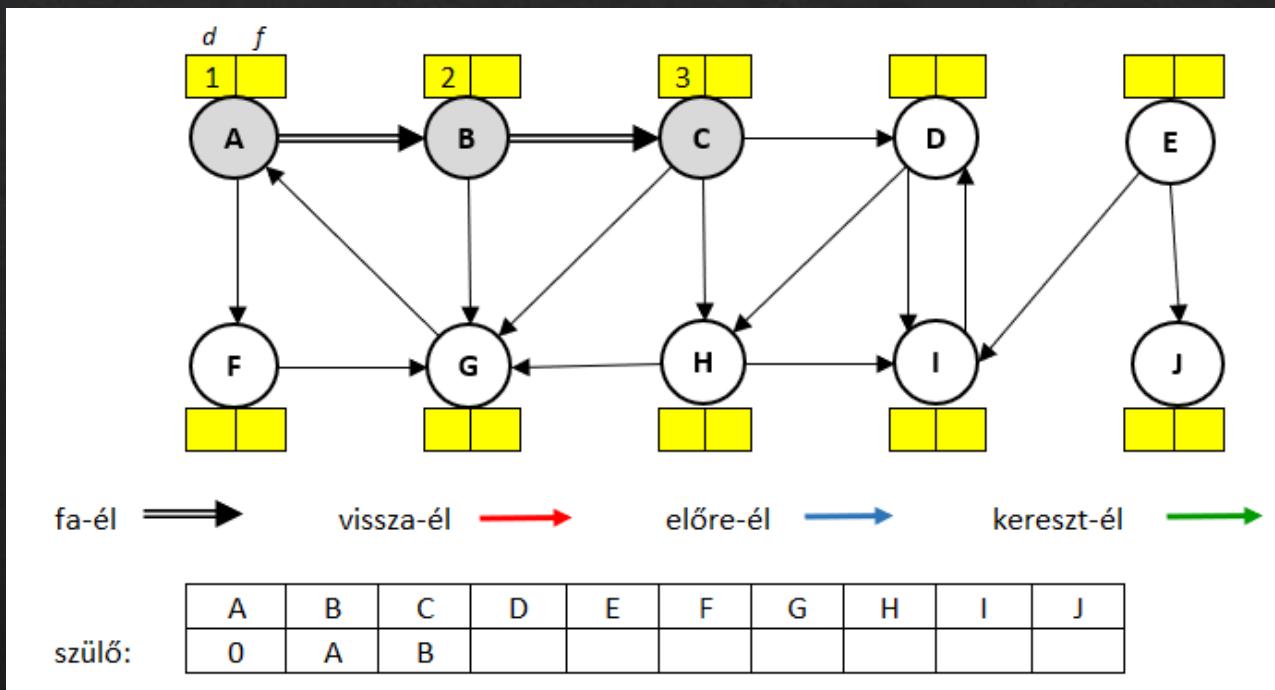
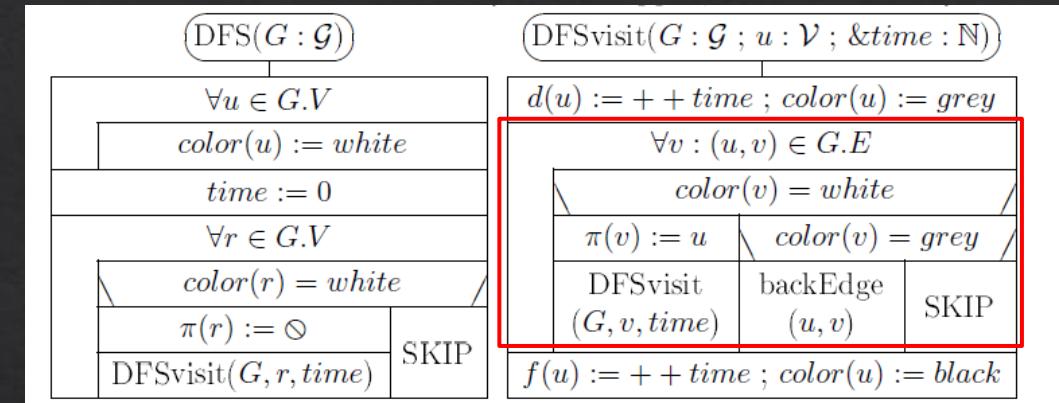
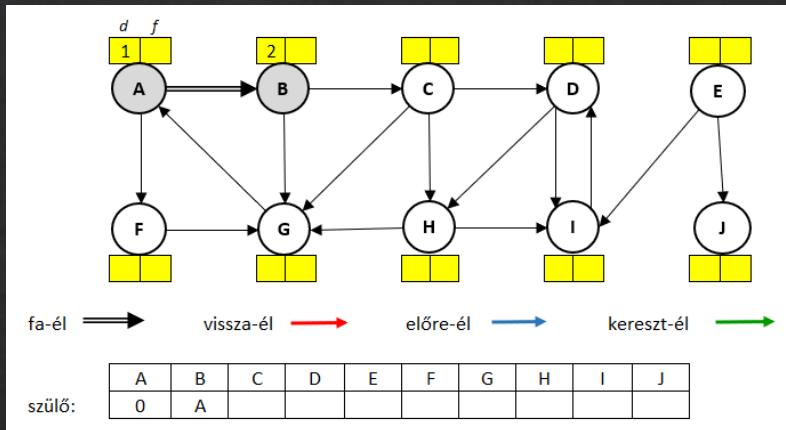


DFS($G : \mathcal{G}$)	DFSvisit($G : \mathcal{G} ; u : \mathcal{V} ; \&time : \mathbb{N}$)	
$\forall u \in G.V$	$d(u) := + + time$; $color(u) := grey$	
$color(u) := white$		
$time := 0$		
$\forall r \in G.V$		
$\backslash \quad color(r) = white /$		
$\pi(r) := \odot$	$\pi(v) := u$	$color(v) = grey$
DFSvisit($G, r, time$)	DFSvisit($G, v, time$)	backEdge (u, v) SKIP
	$f(u) := + + time$; $color(u) := black$	

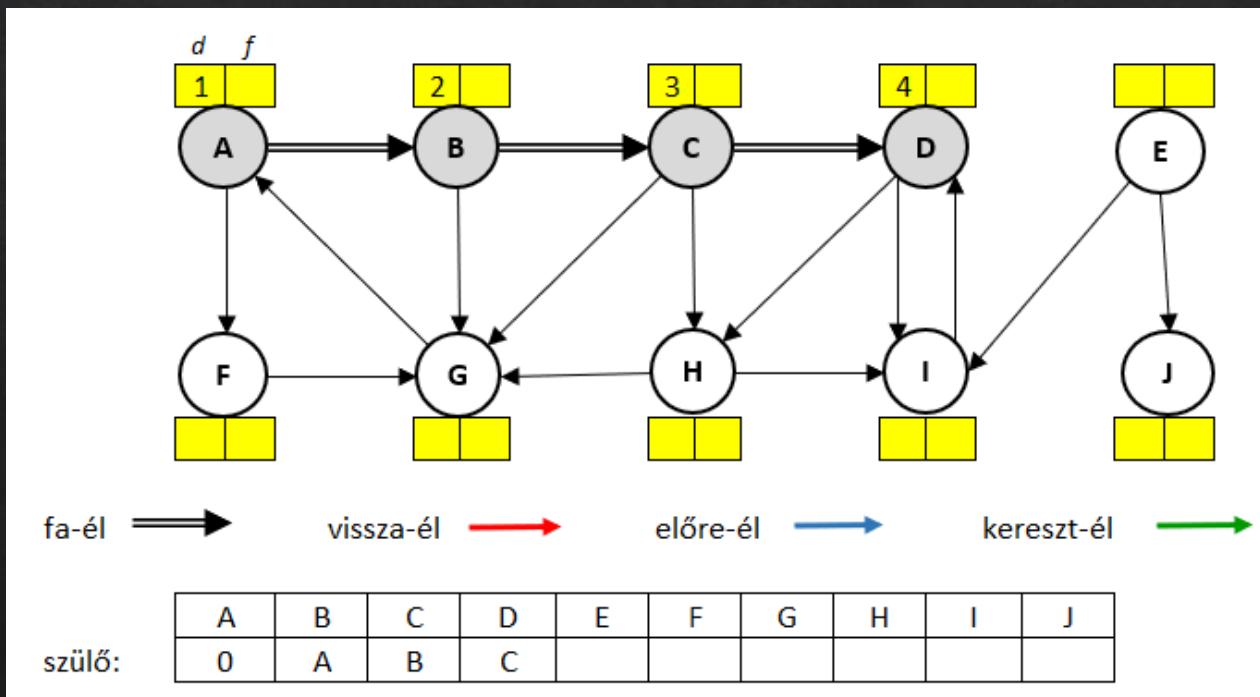
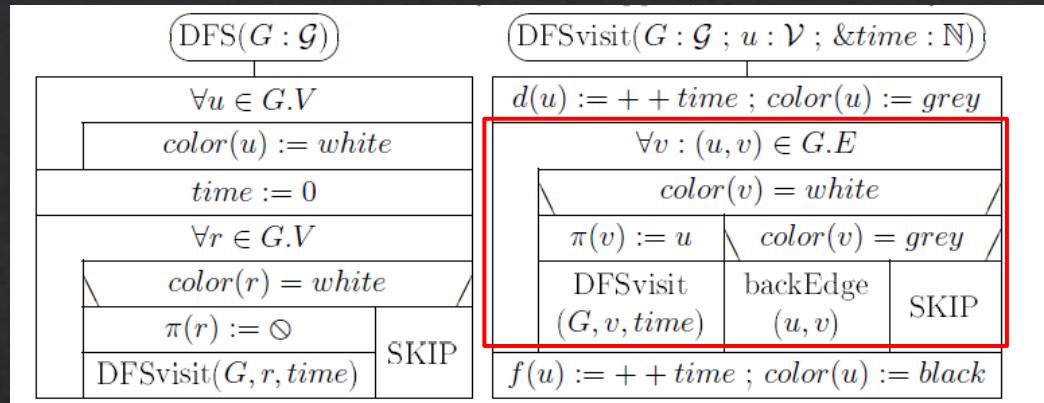
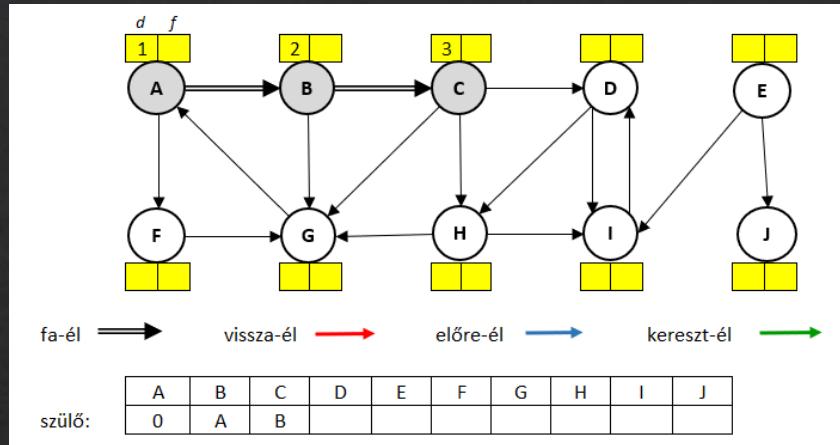
- ❖ Első fehér csúcs: A
- ❖ $\pi(A) := 0$
- ❖ Belép a DFSvisit algoritmusba,
- ❖ Idő megnövekszik, $d(A) := 1$
- ❖ $color(A) := grey$



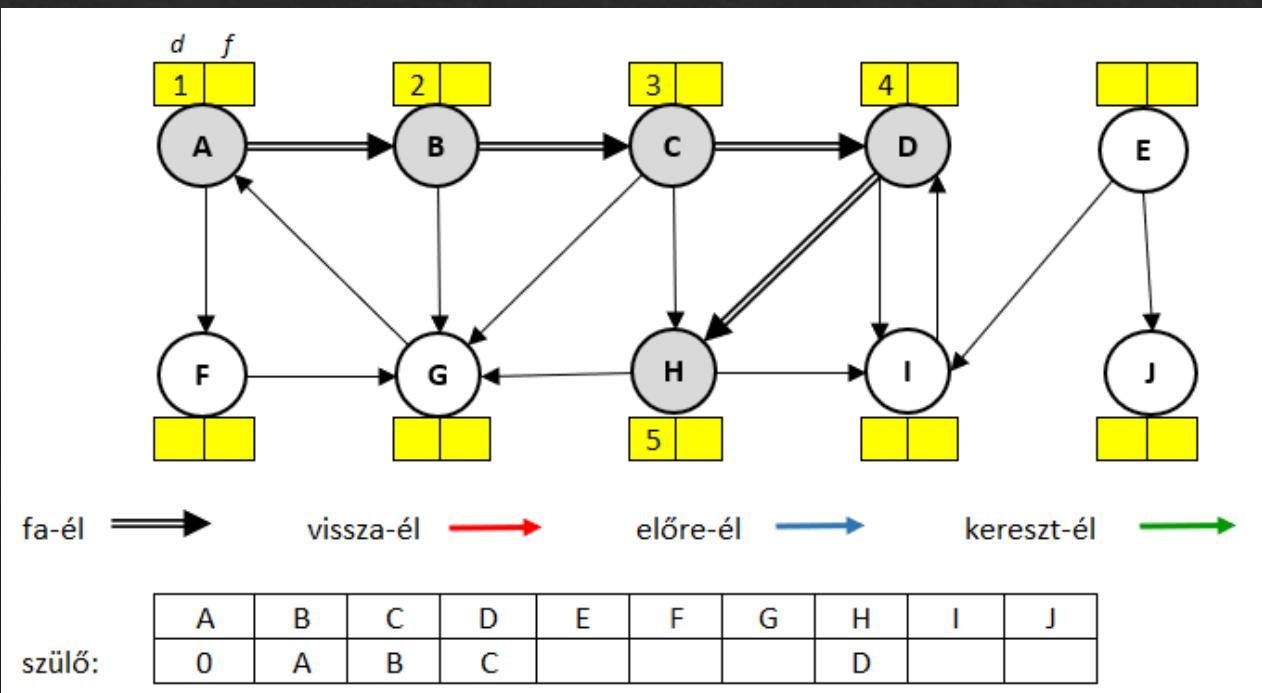
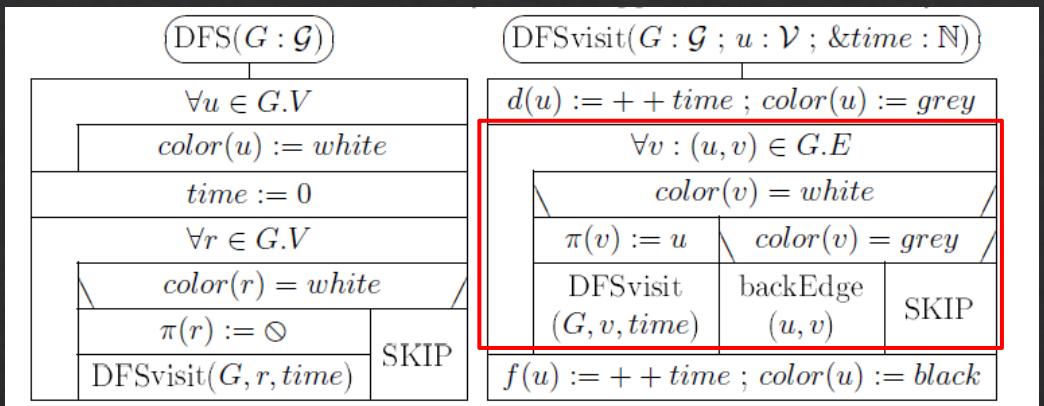
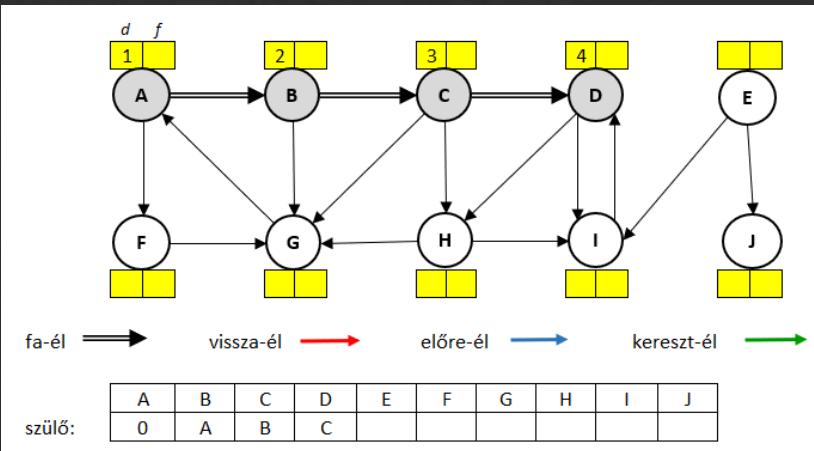
- ❖ Ábécé szerinti első szomszéd: B, fehér.
- ❖ $\pi(B) := A$
- ❖ Rekuzívan hívja DFSvisit algoritmust B csúcsra,
- ❖ Idő megnövekszik, $d(B) := 2$
- ❖ $\text{color}(B) := \text{grey}$



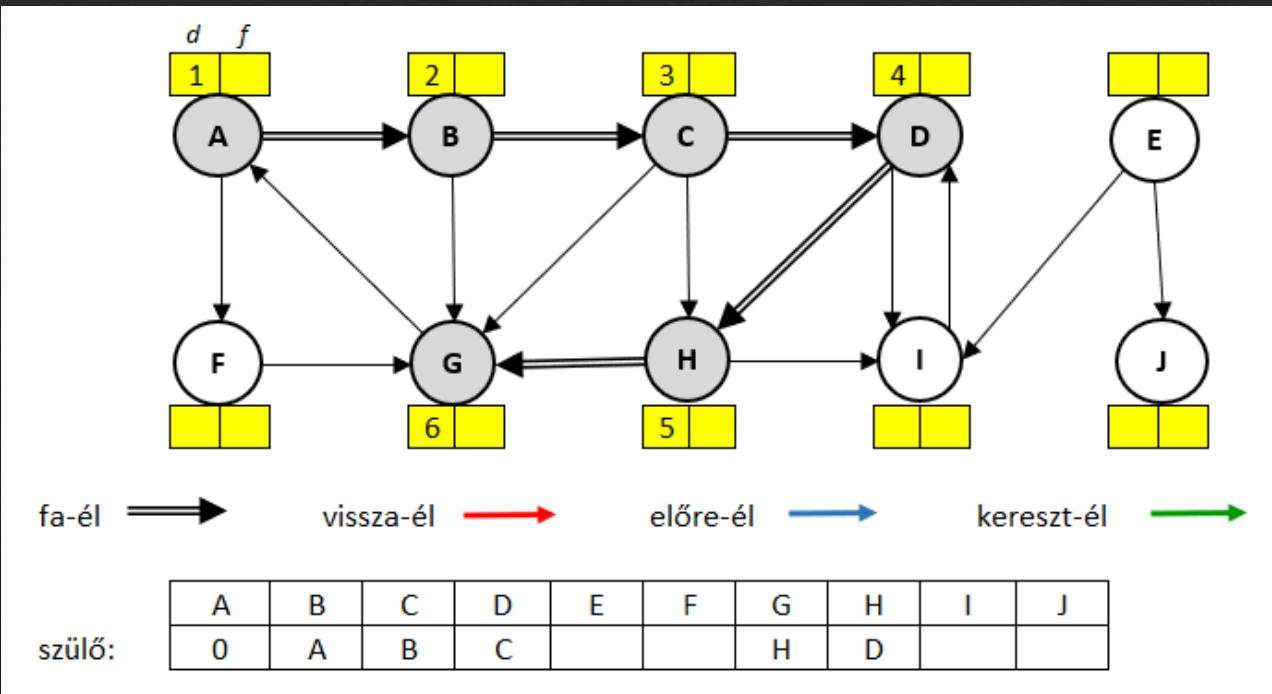
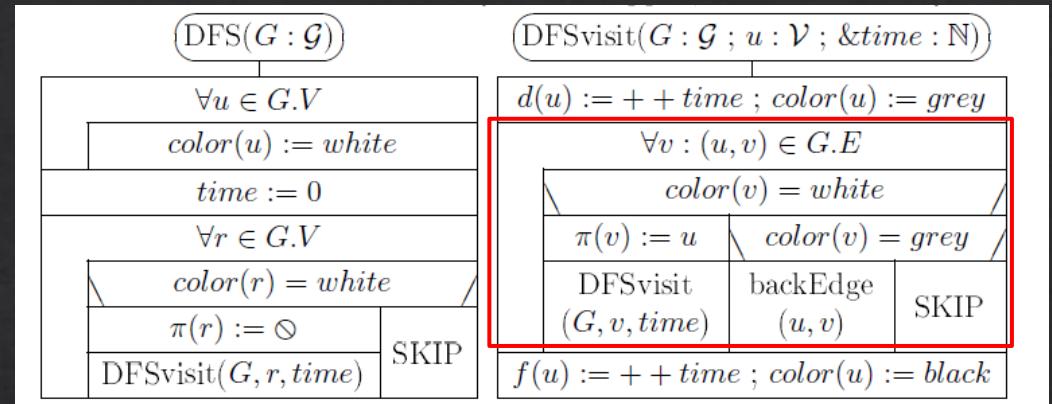
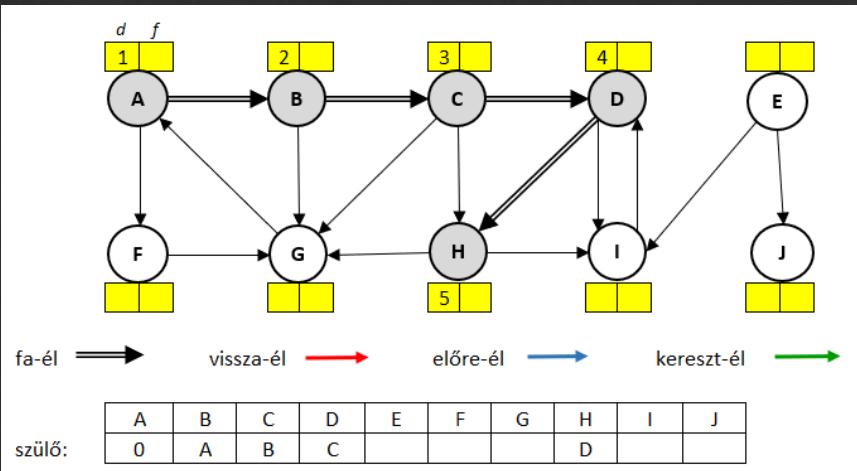
- ❖ Ábécé szerinti első szomszéd: C, fehér csúcs
- ❖ $\pi(C) := B$
- ❖ Rekuzívan hívja DFSvisit algoritmust C csúcsra,
- ❖ Idő megnövekszik, $d(C) := 3$
- ❖ $\text{color}(C) := \text{grey}$



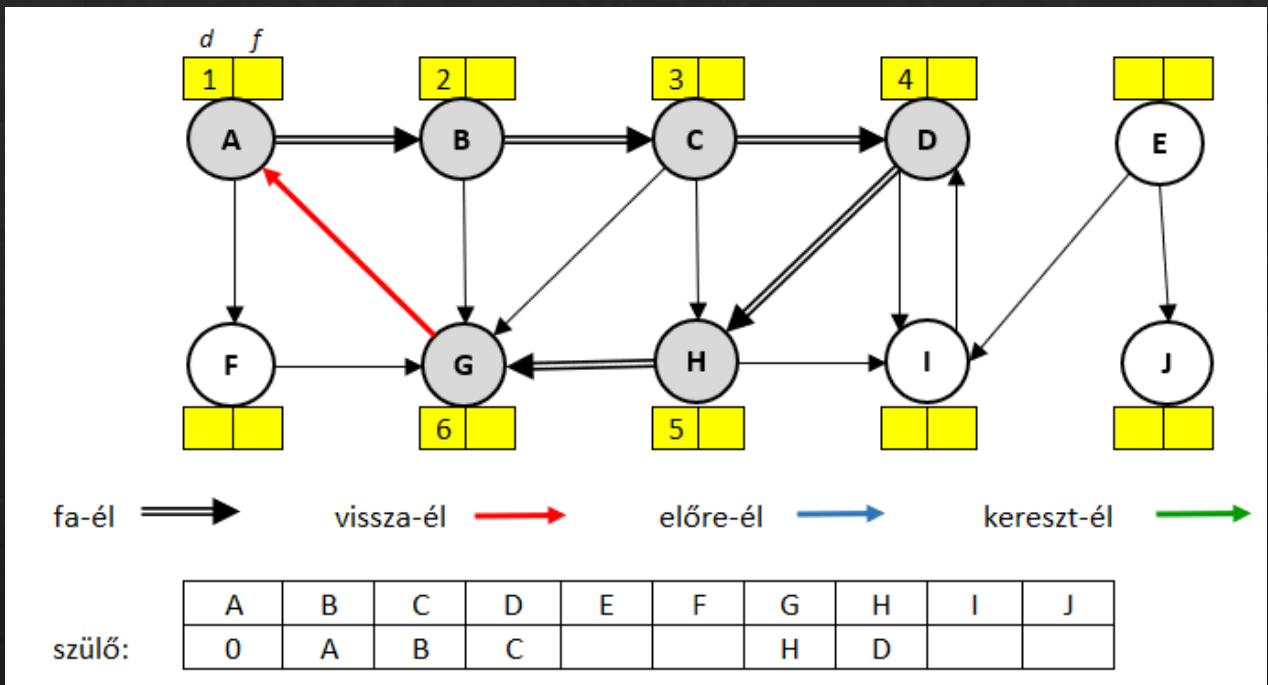
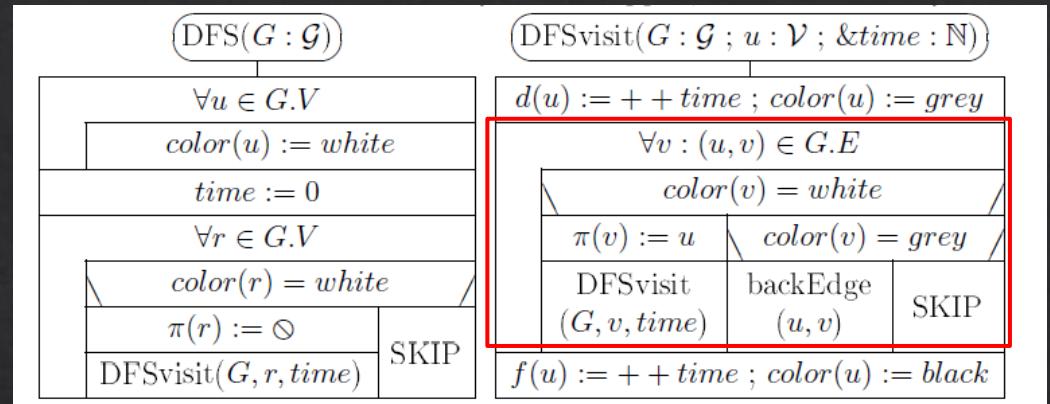
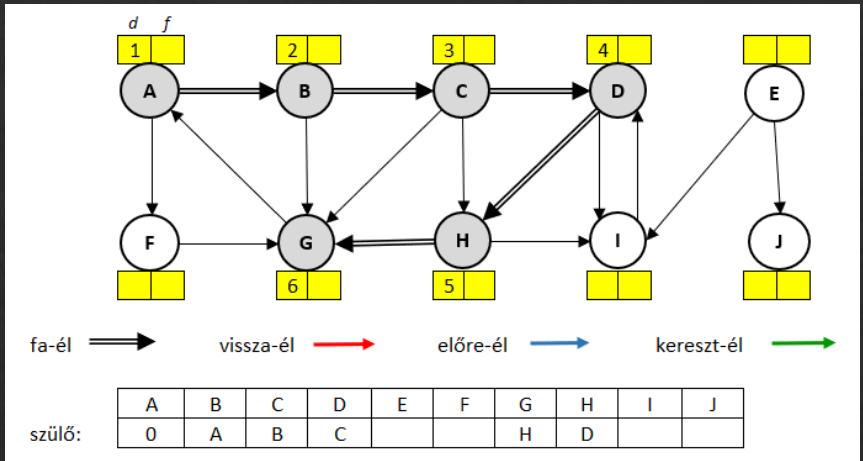
- ❖ C első szomszédja: D, fehér
- ❖ $\pi(D) := C$
- ❖ Rekuzívan hívja DFSvisit algoritmust D csúcsra,
- ❖ Idő megnövekszik, $d(D) := 4$
- ❖ $color(D) := \text{grey}$



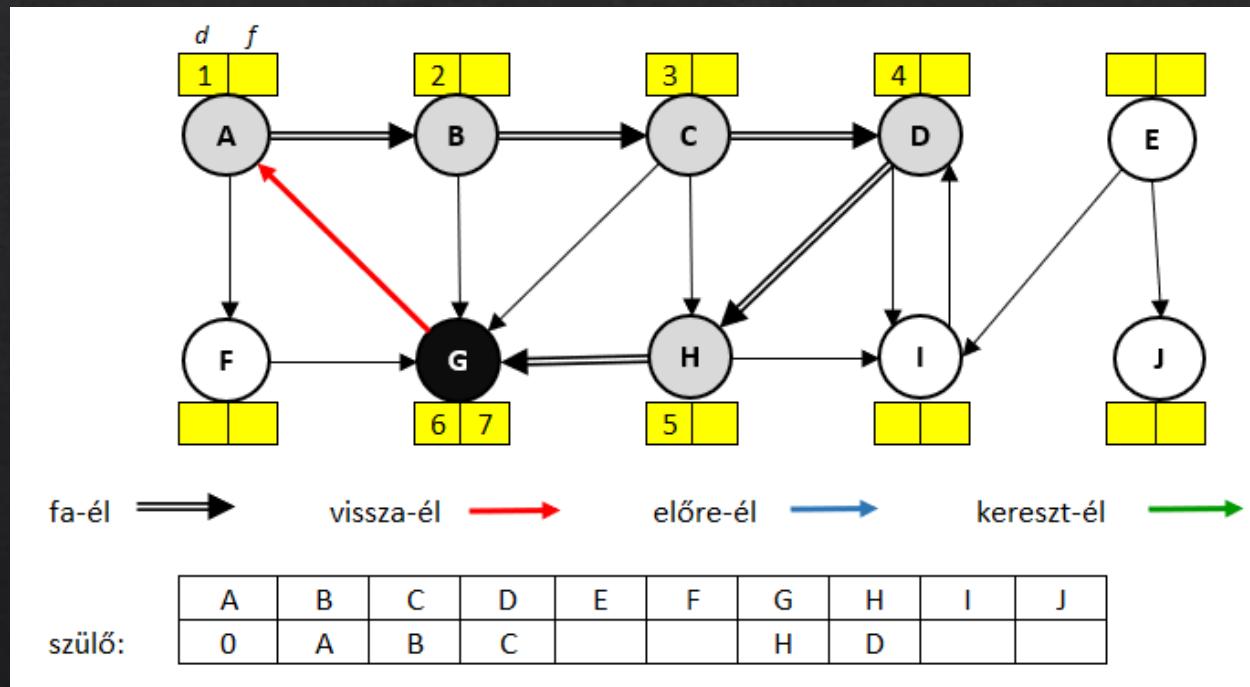
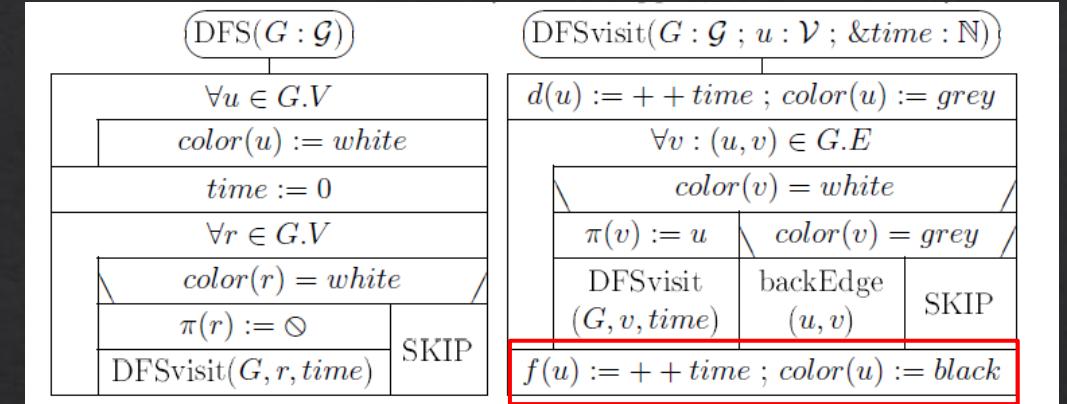
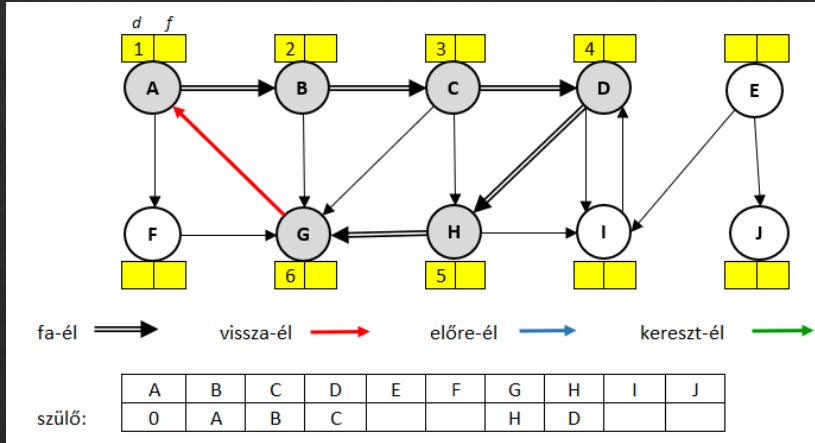
- ❖ D-ből H csúcsba megy
- ❖ $\pi(H) := D$
- ❖ Rekuzívan hívja DFSvisit algoritmust H csúcsra,
- ❖ Idő megnövekszik, $d(H) := 5$
- ❖ $\text{color}(H) := \text{grey}$



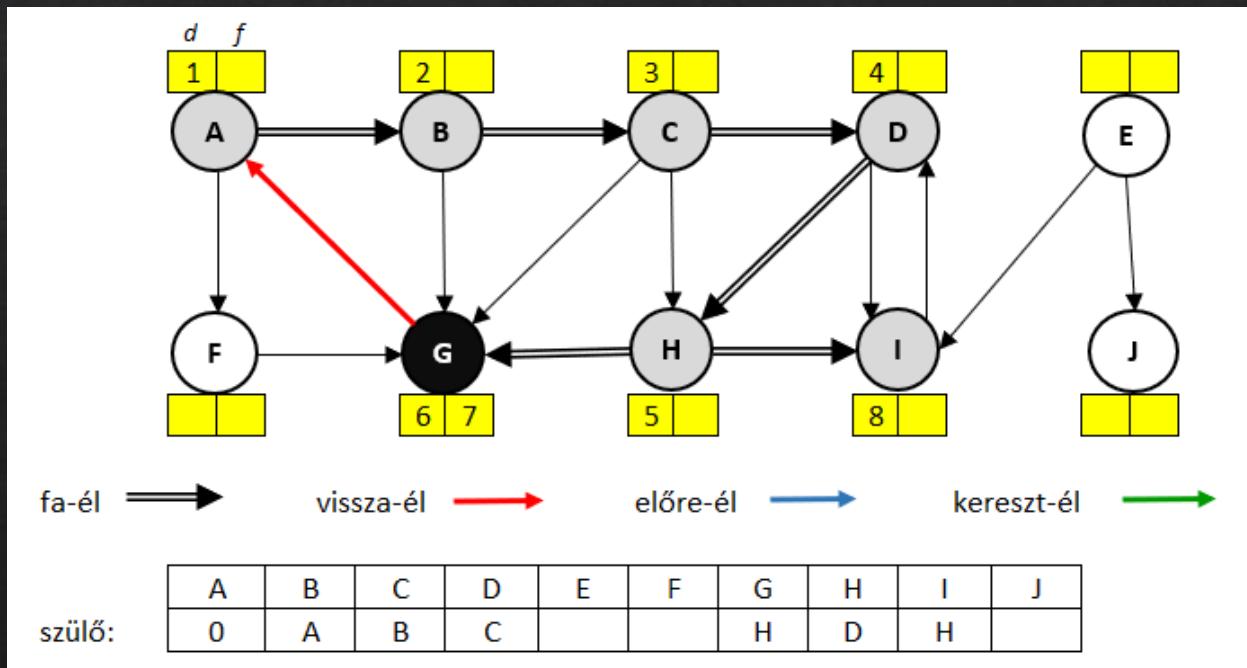
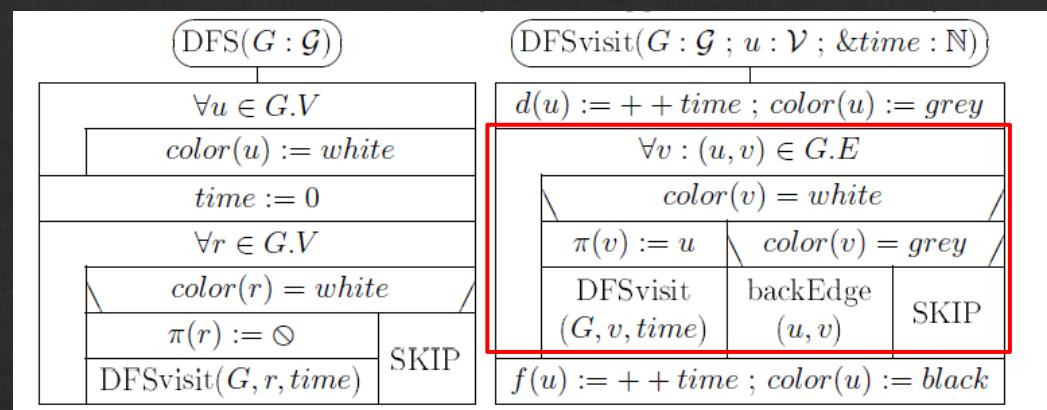
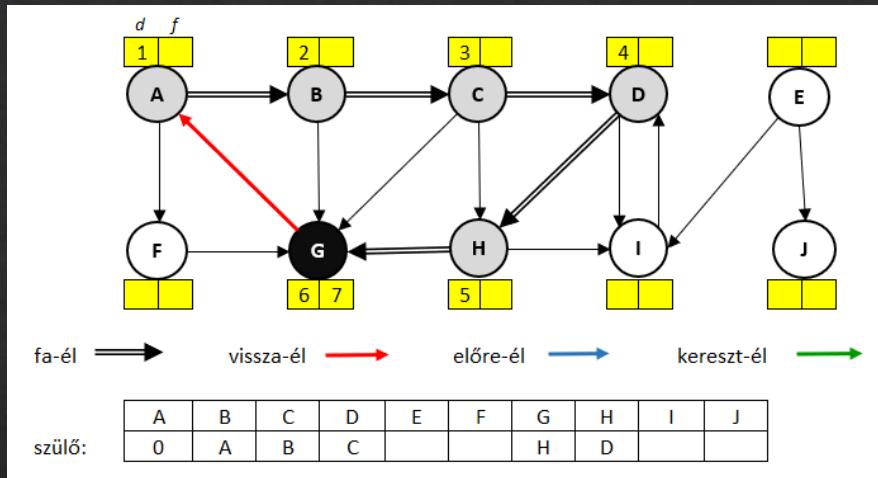
- ❖ H-ból G csúcsba megy
- ❖ $\pi(G) := H$
- ❖ Rekuzívan hívja DFSvisit algoritmust G csúcsra,
- ❖ Idő megnövekszik, $d(G) := 6$
- ❖ $\text{color}(G) := \text{grey}$



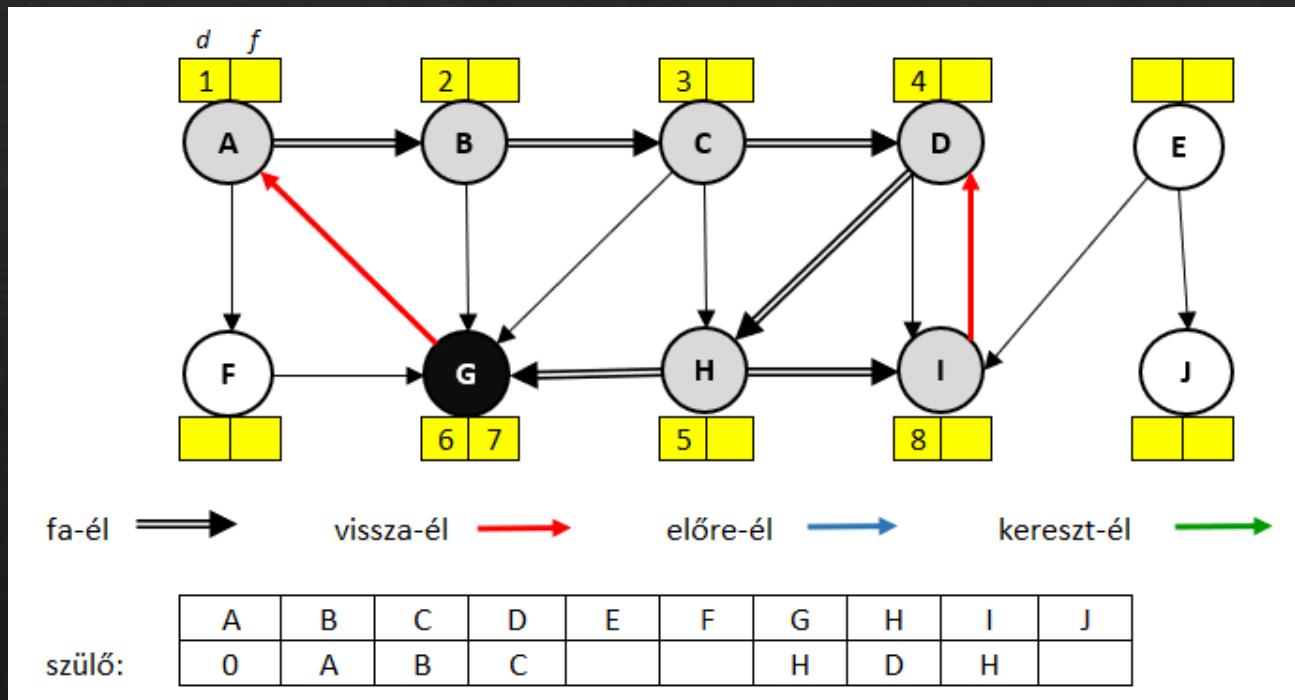
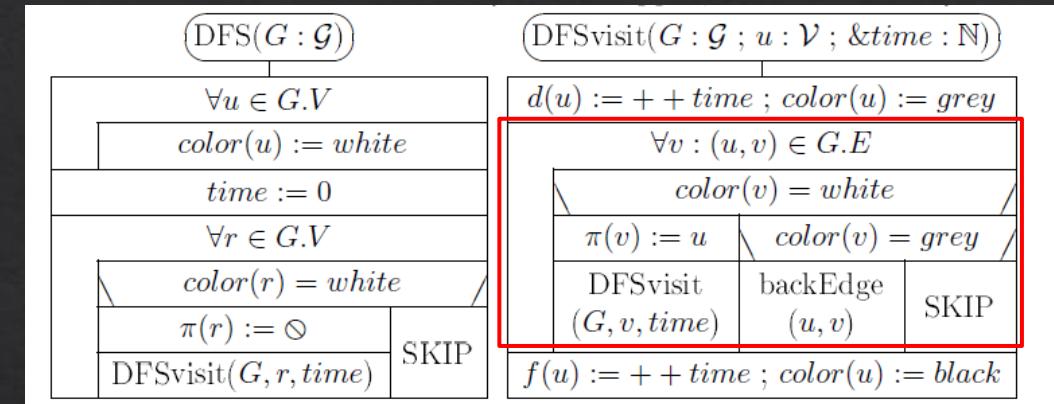
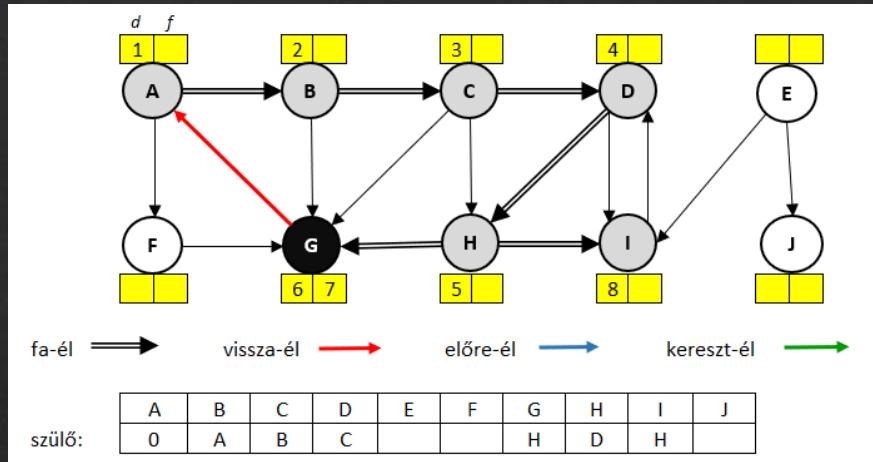
- ◆ G szomszédja az A, de az már szürke csúcs
- ◆ Vissza-élt talált



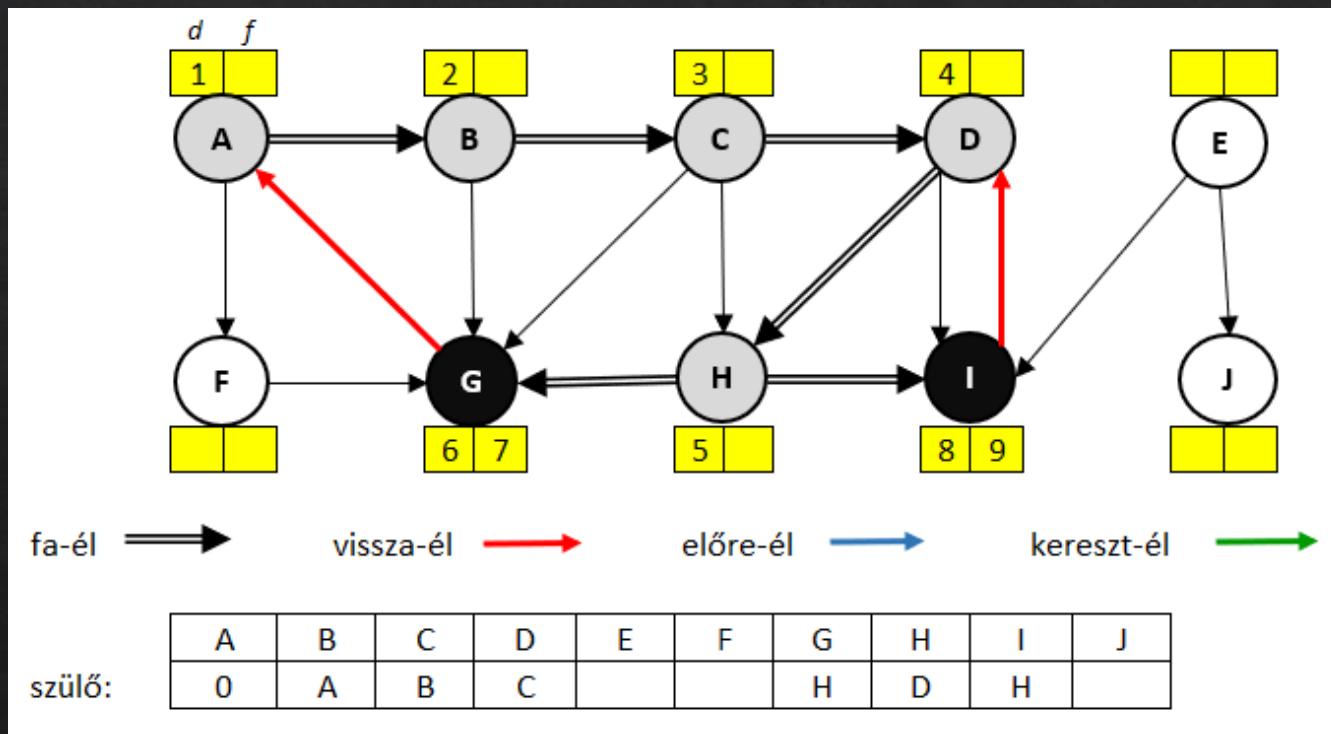
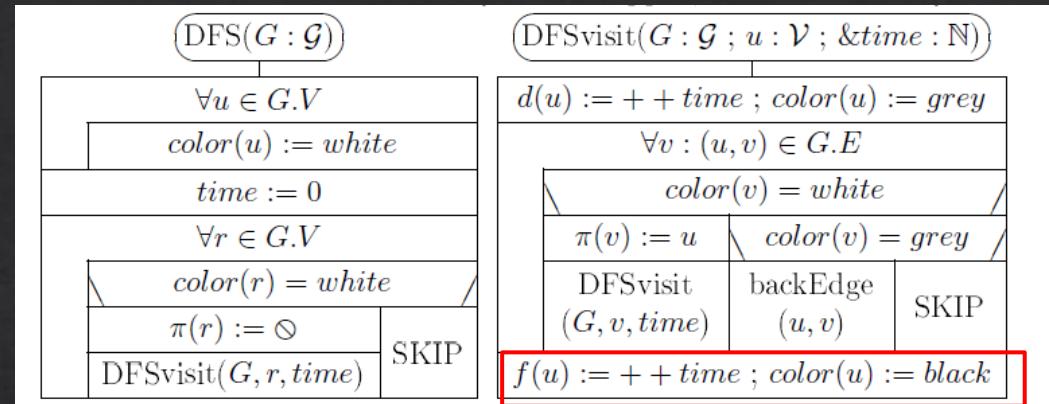
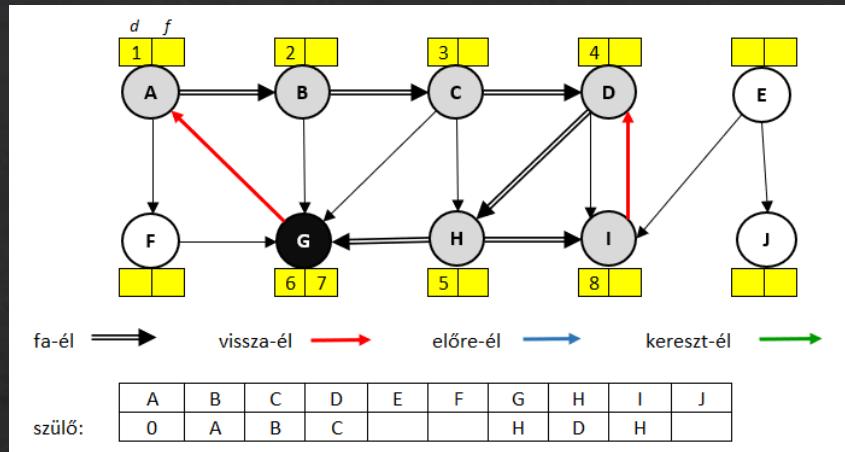
- ❖ G -nek nincs több szomszédja
- ❖ Feldolgozása befejeződött
- ❖ Idő növekszik,
- ❖ $f(G) := 7$
- ❖ $color(G) := black$



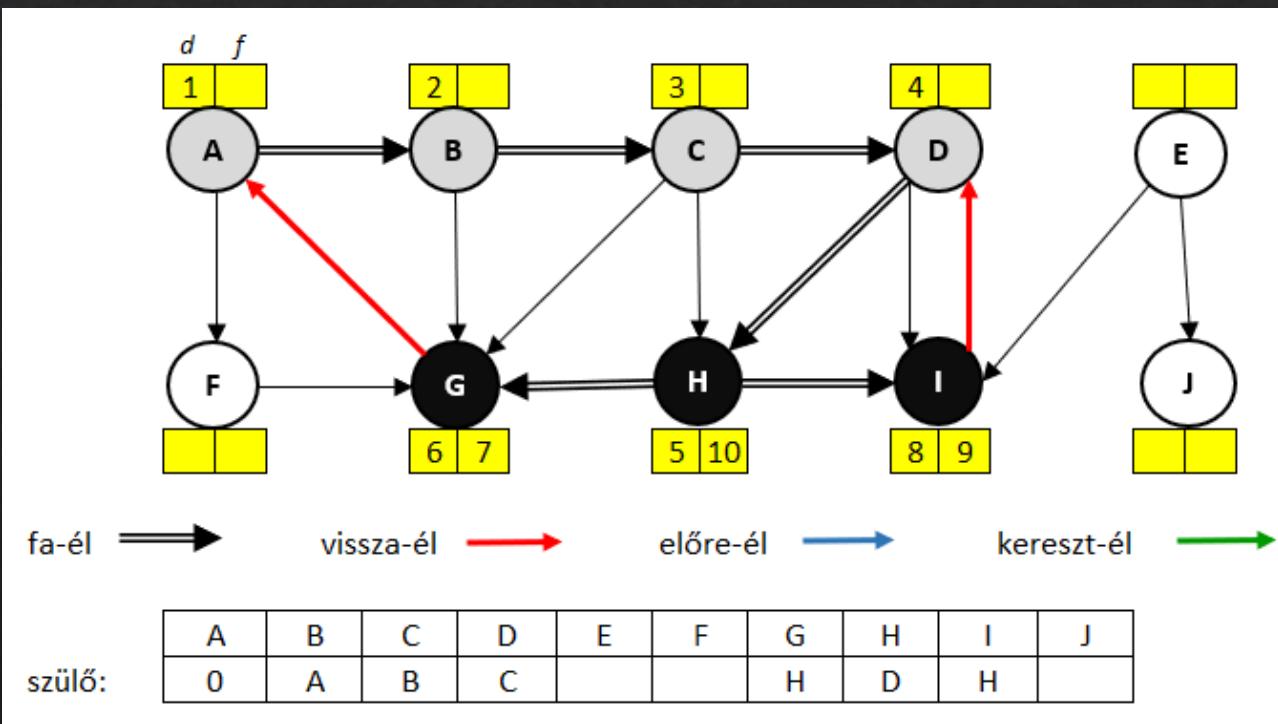
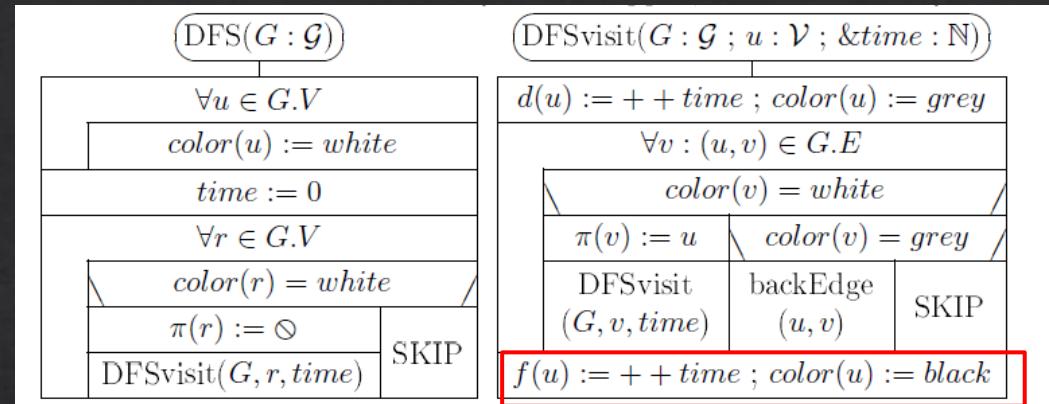
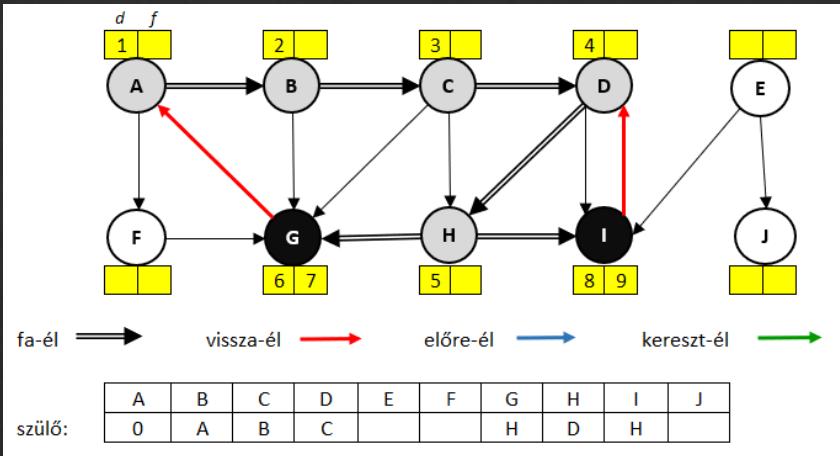
- ◆ G szülőjének, H-nak a feldolgozása folytatódik.
- ◆ I a szomszédja, és fehér
- ◆ $\pi(I) := H$
- ◆ $d(I) := 8$
- ◆ $\text{color}(I) := \text{grey}$



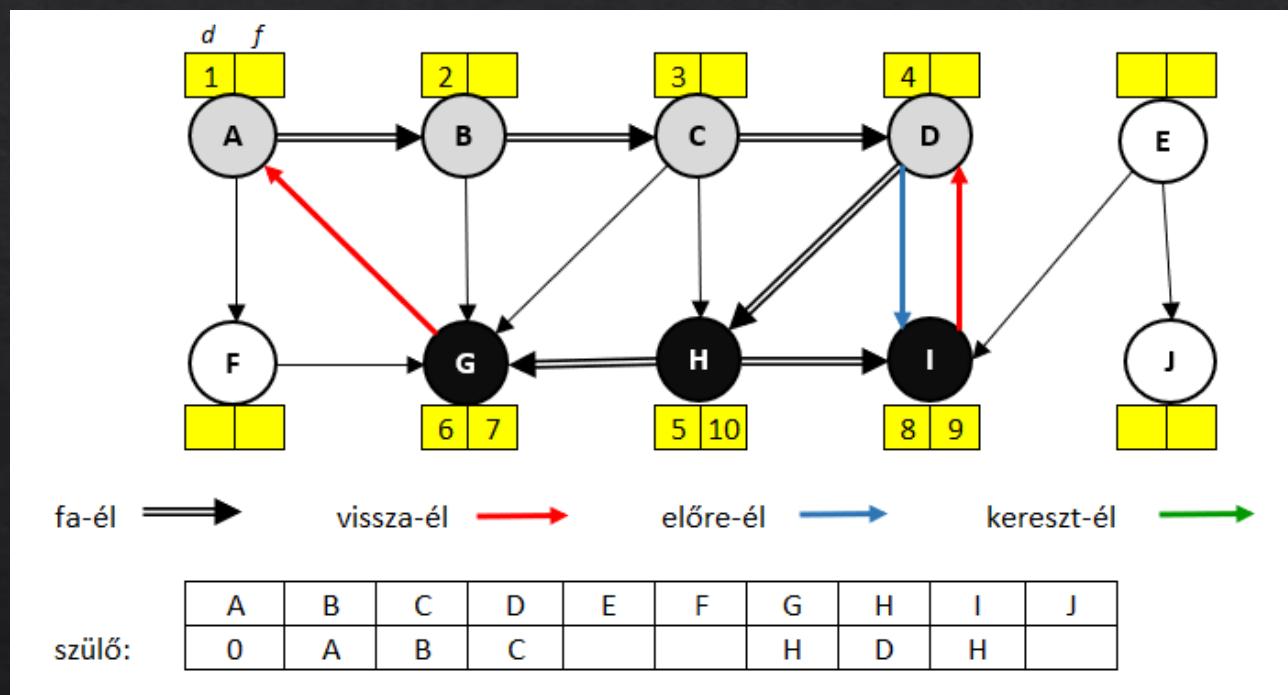
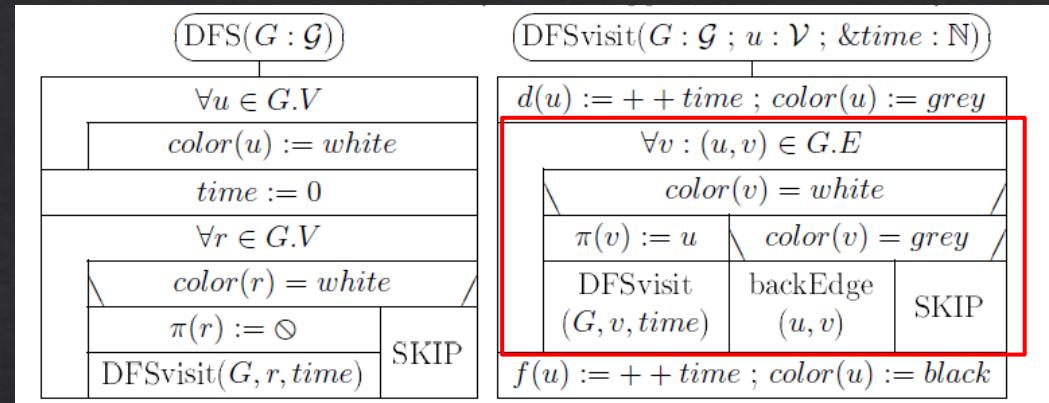
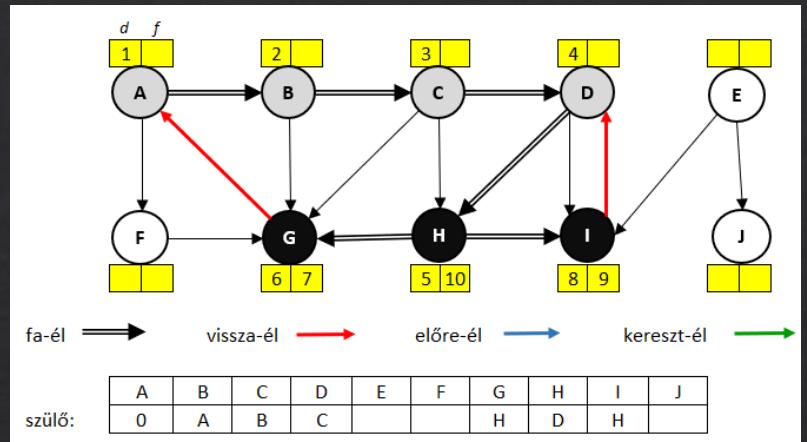
- ♦ I szomszédja D, ami szürke színű.
- ♦ Vissza-élt találtunk



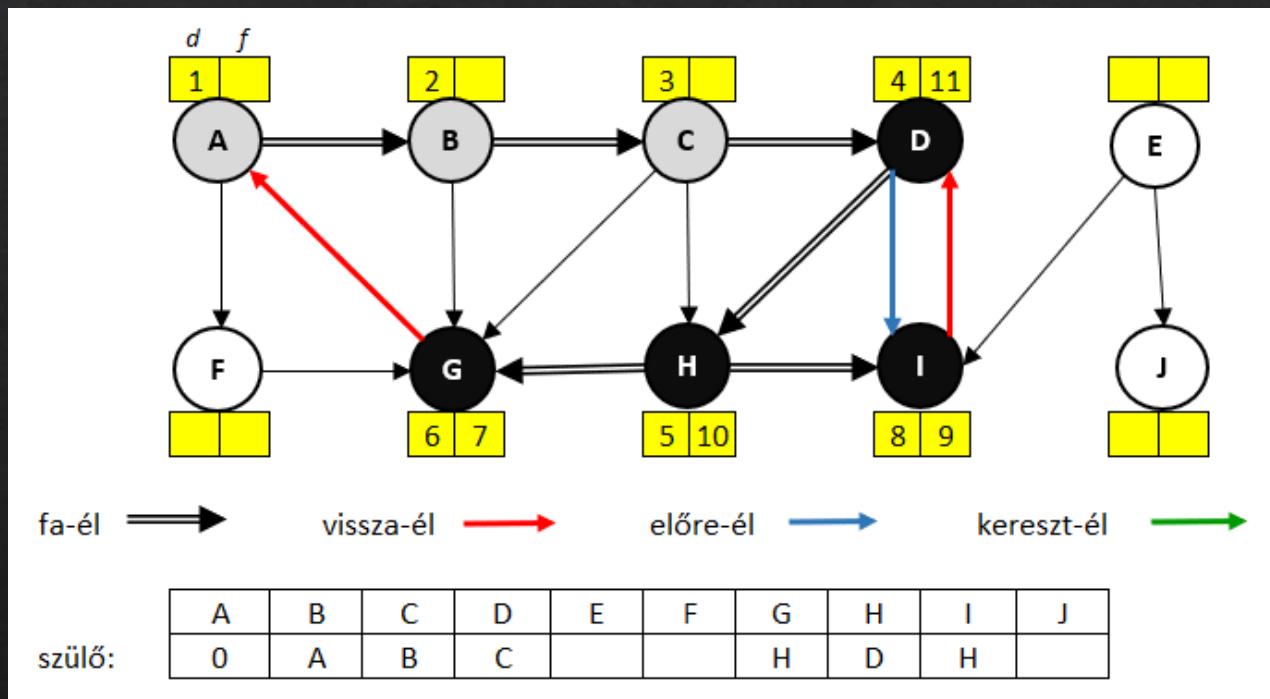
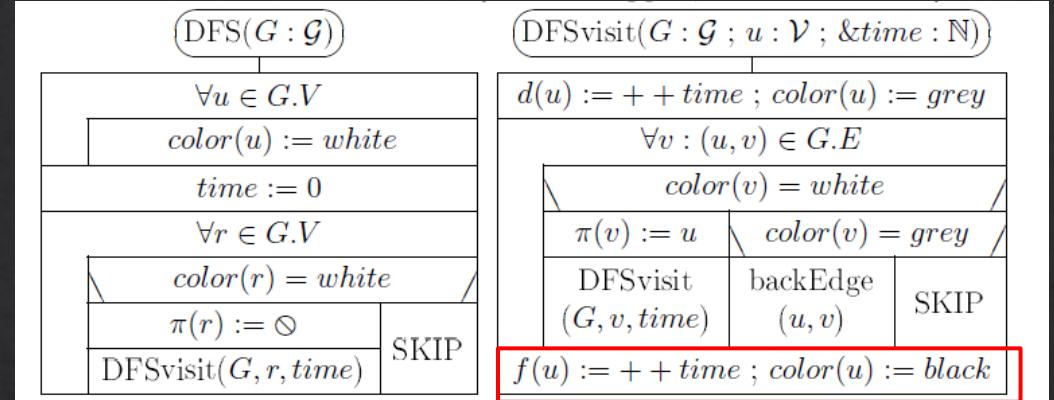
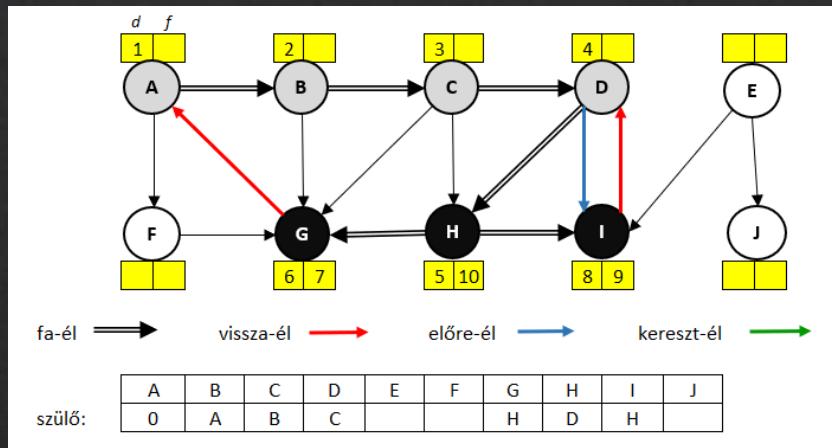
- ❖ I is teljesen feldolgozott lett.
- ❖ Idő növekszik,
- ❖ $f(I) := 9$
- ❖ $\text{color}(I) := \text{black}$



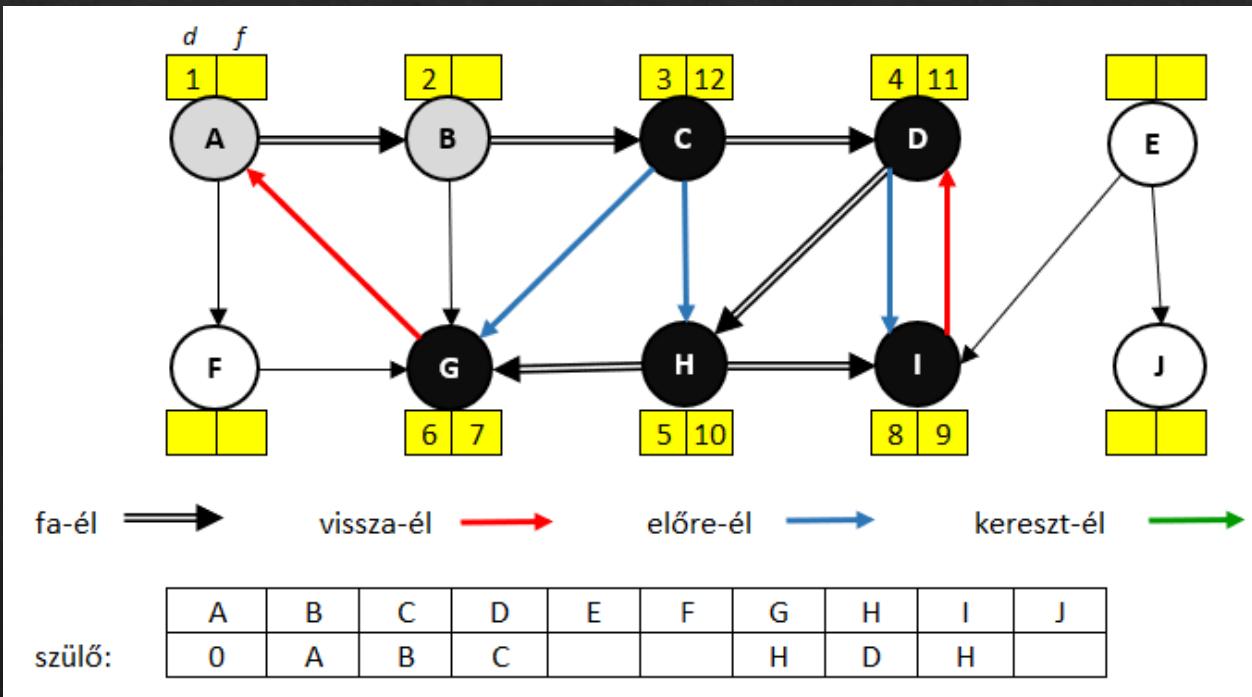
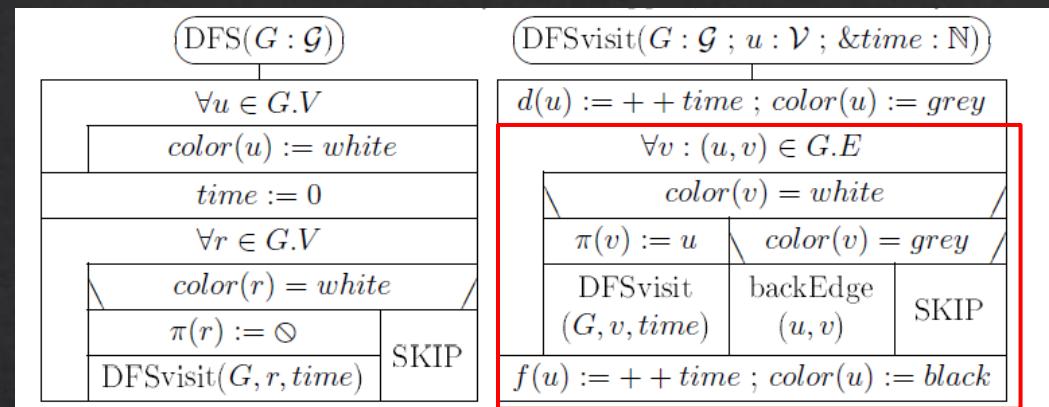
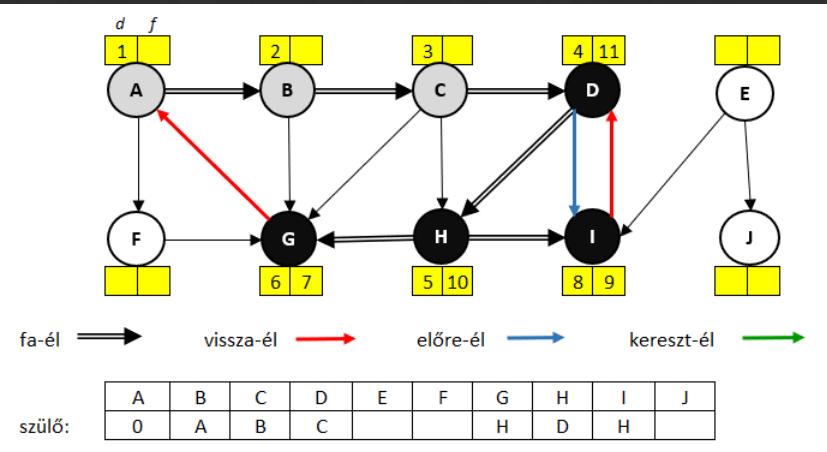
- ◆ H is kész csúcs lett
- ◆ Idő növekszik,
- ◆ $f(H) := 10$
- ◆ $\text{color}(H) := \text{black}$



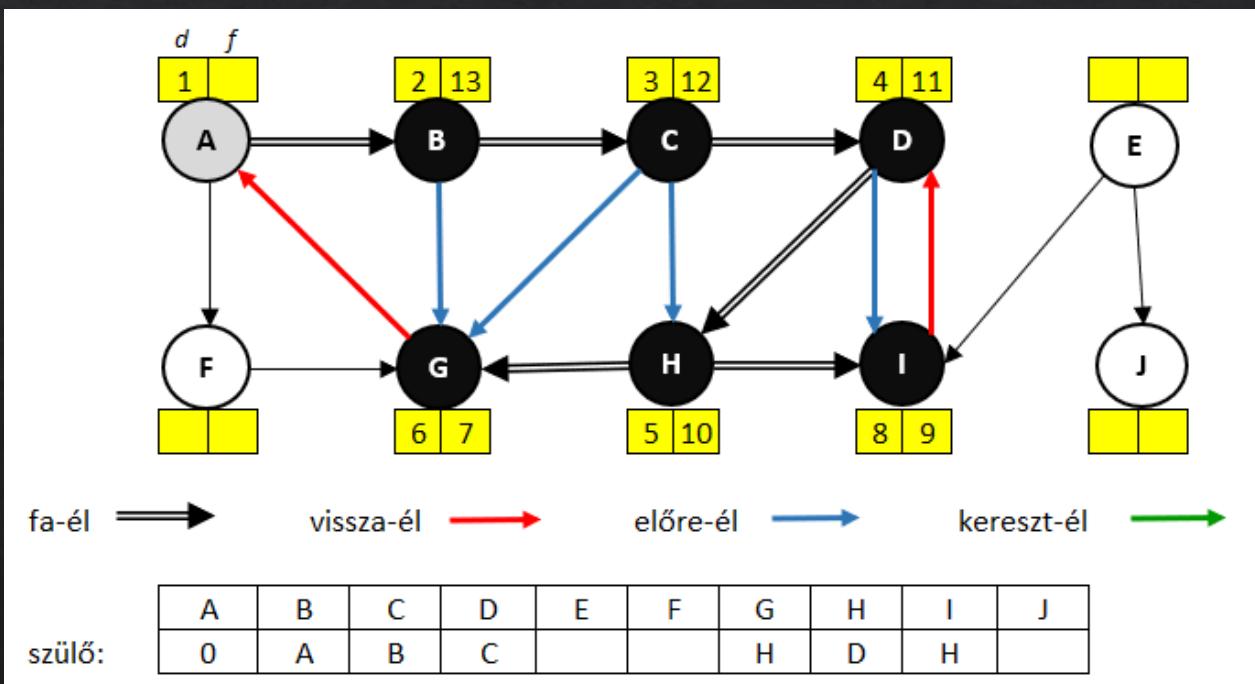
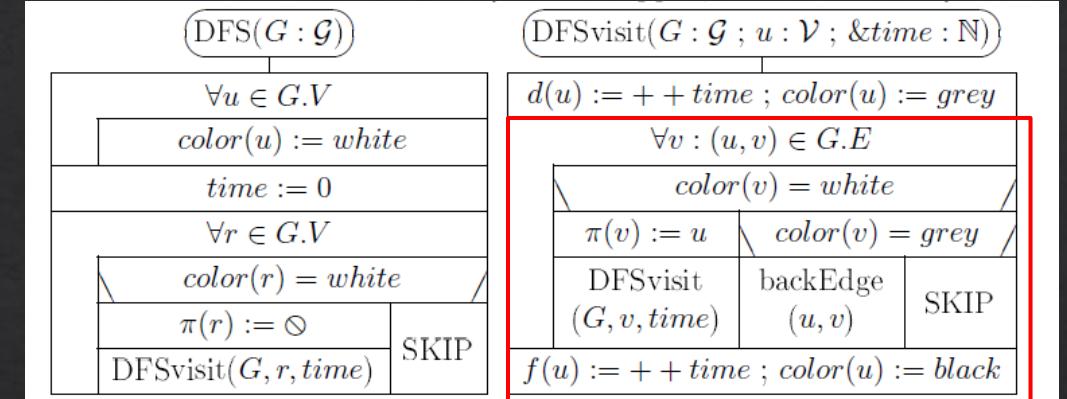
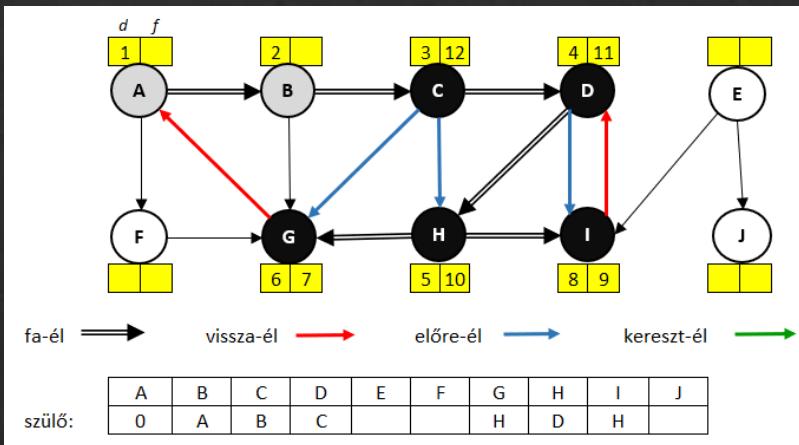
- ◆ D-be tért vissza a rekurzió
- ◆ I a szomszédja, fekete, $d(D) < d(I)$, tehát előre-élt talált



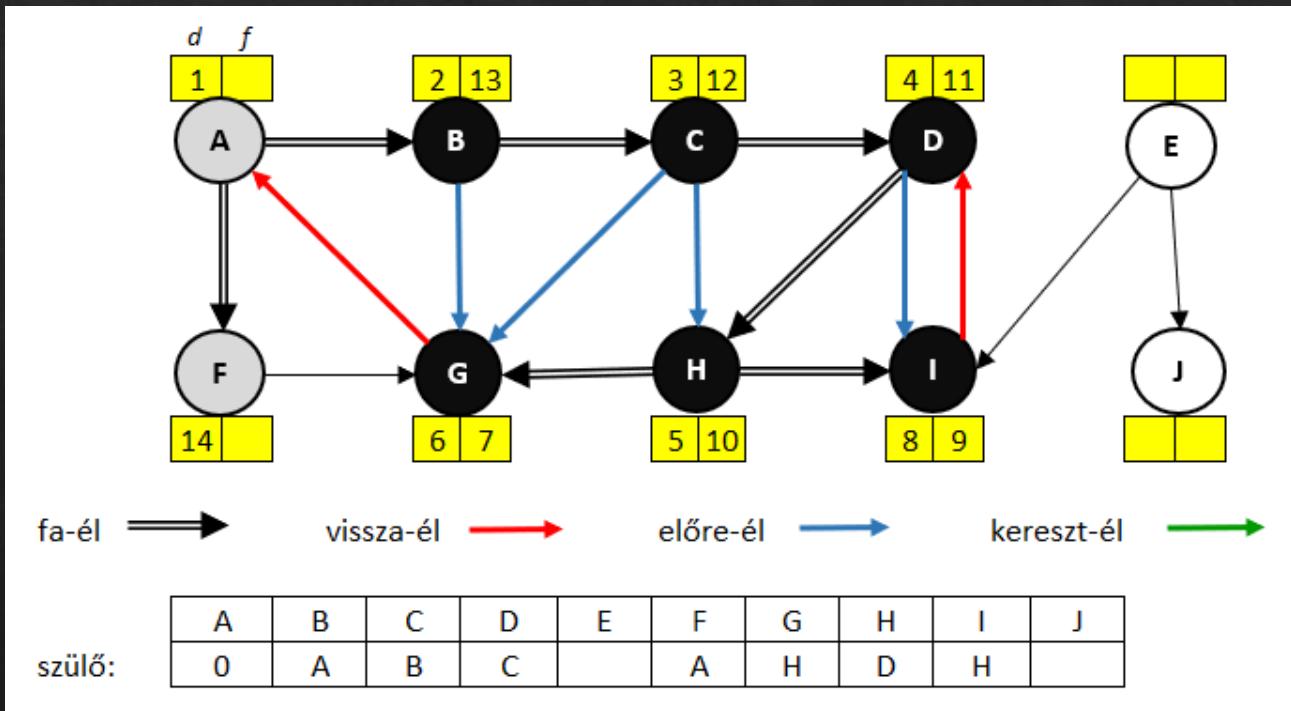
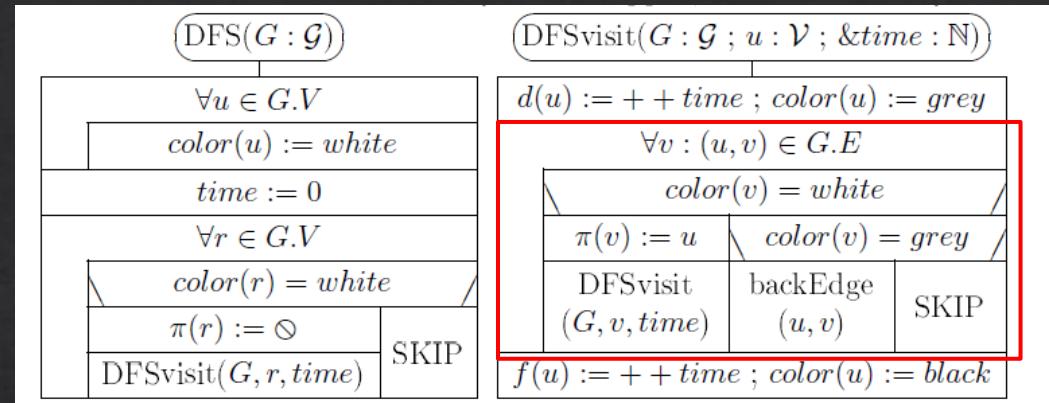
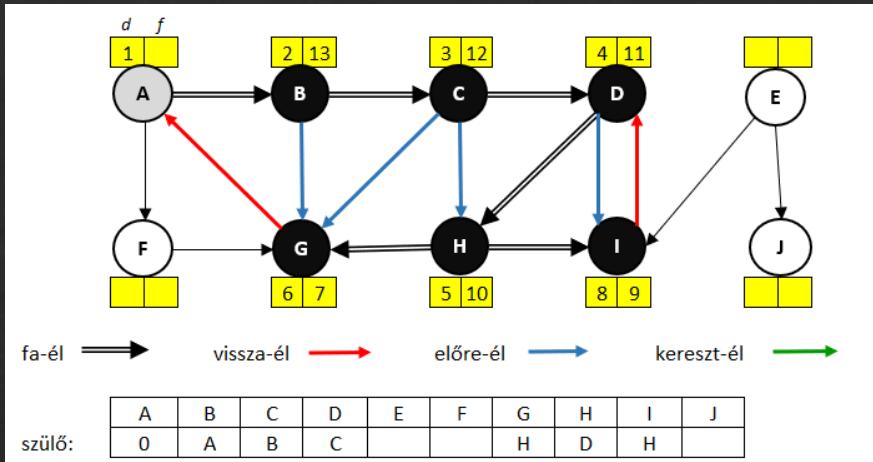
- ◆ D is kész csúcs lett
- ◆ $f(D) := 11$
- ◆ $\text{color}(D) := \text{fekete}$



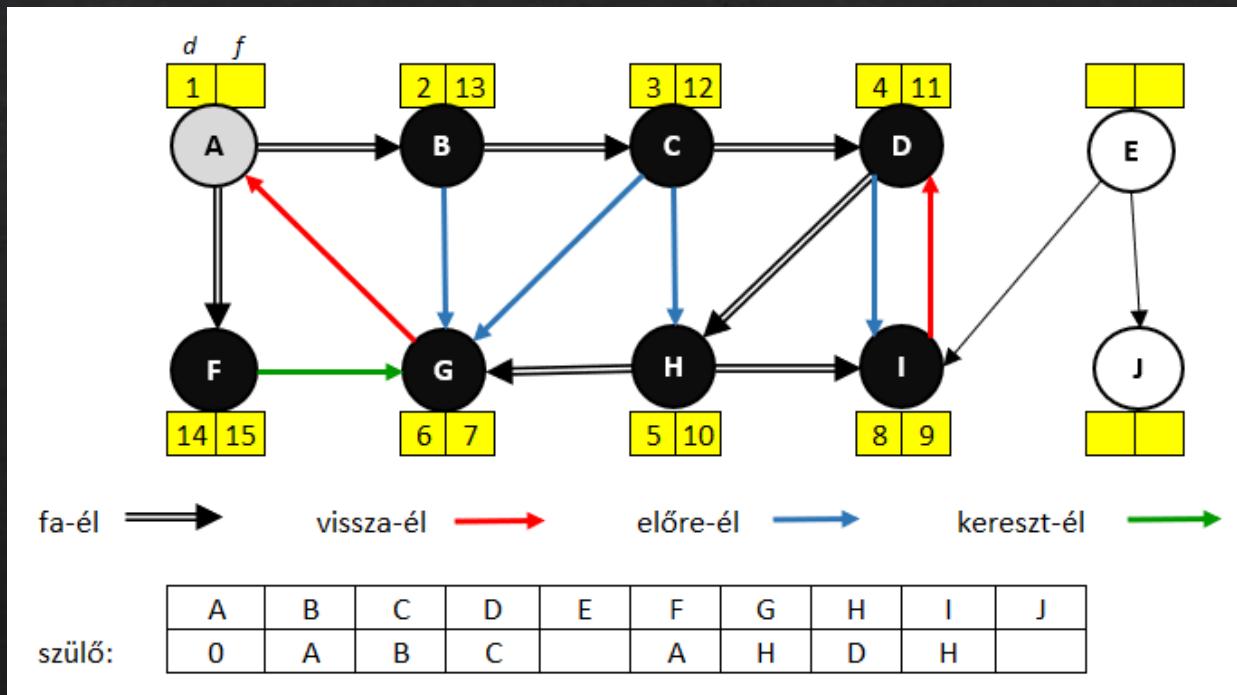
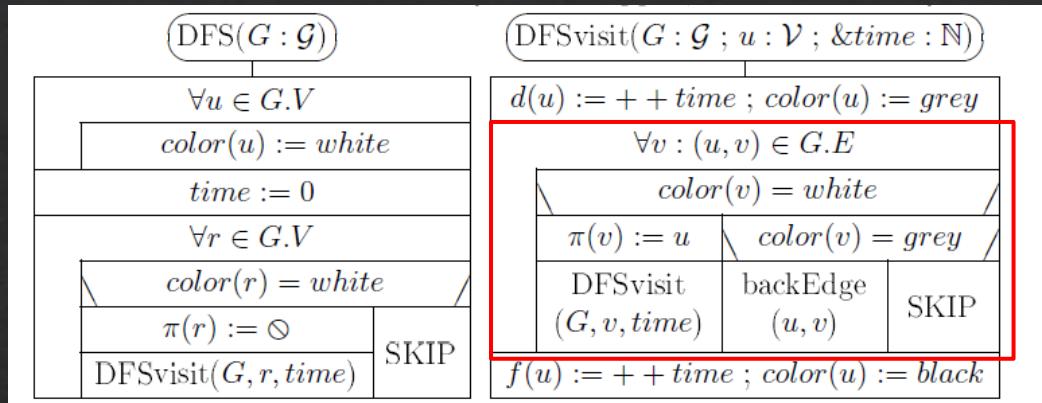
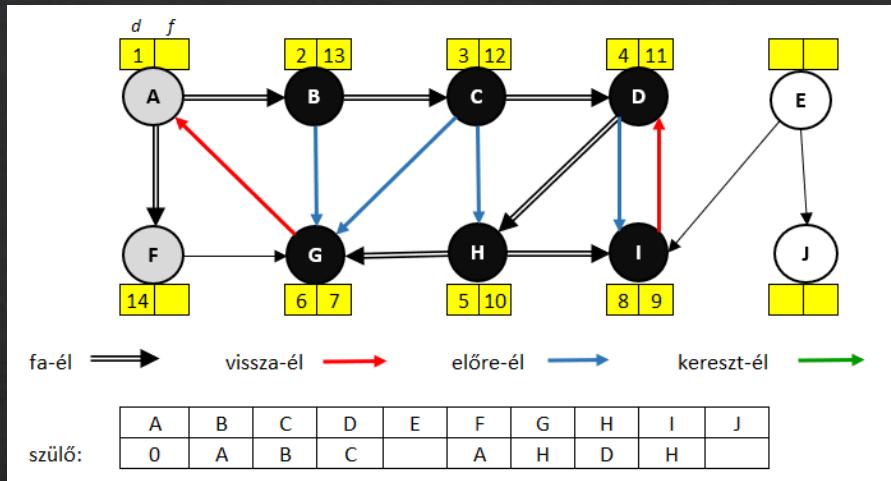
- ◆ C-be tért vissza a rekurzió
- ◆ G,H fekete színű szomszéd,
- ◆ Két újabb előre élt talált.
- ◆ C feldolgozása befejeződött



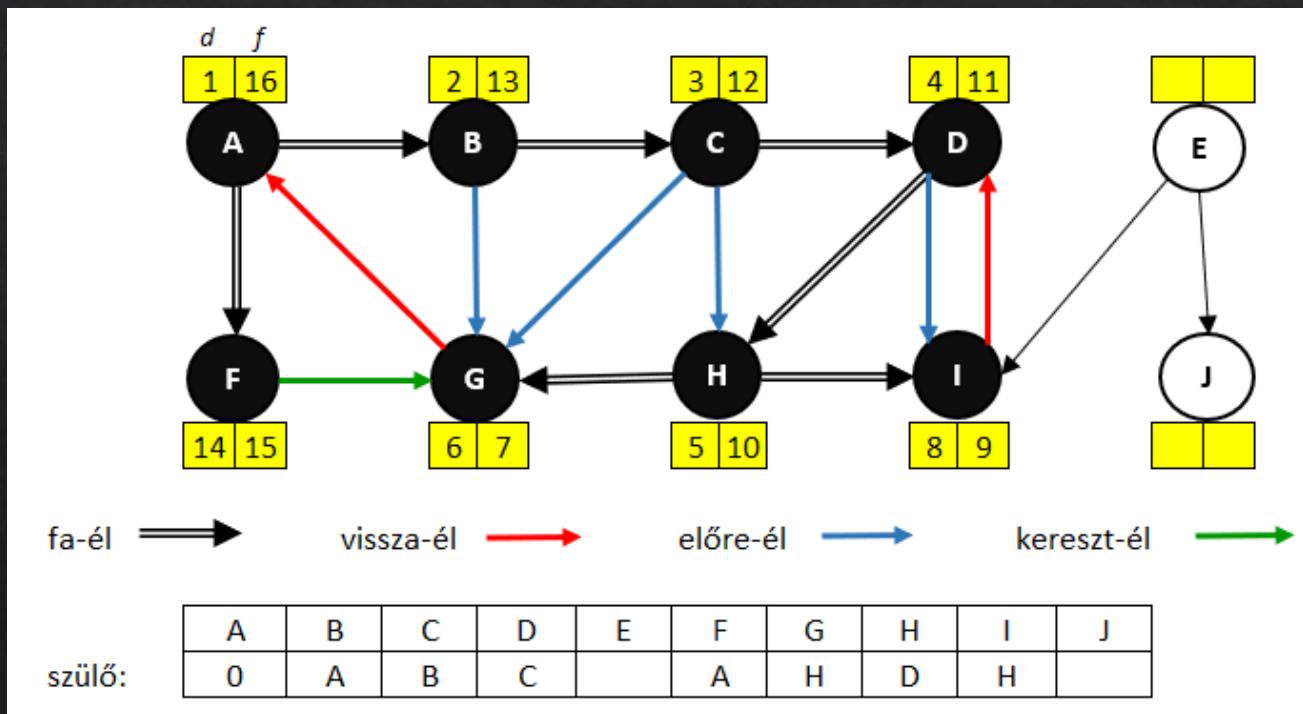
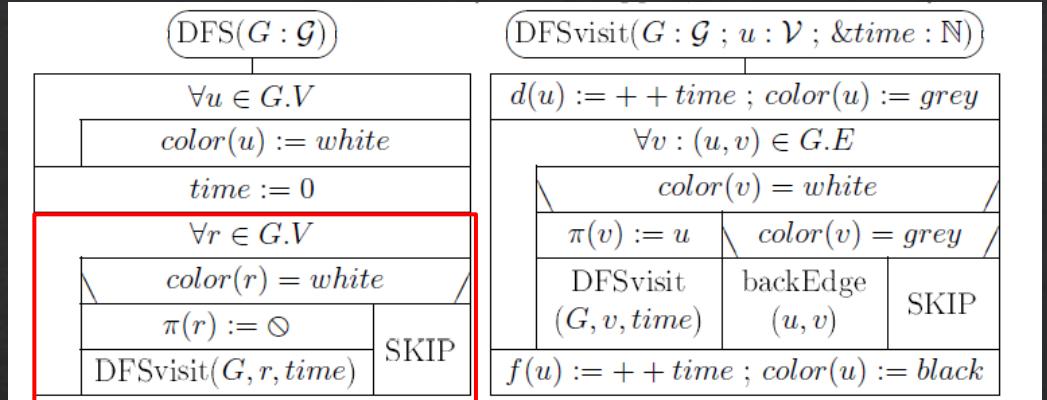
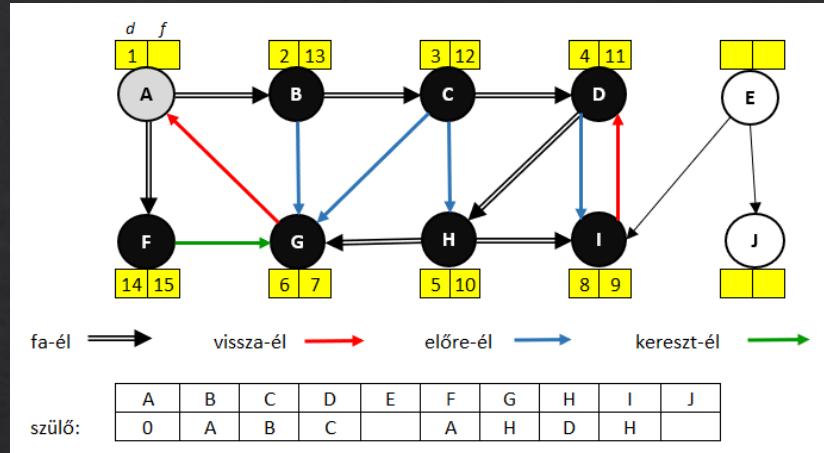
- ◆ B-be tért vissza a rekurzió
- ◆ G fekete színű szomszéd,
- ◆ Újabb előre élt talált.
- ◆ B feldolgozása befejeződött



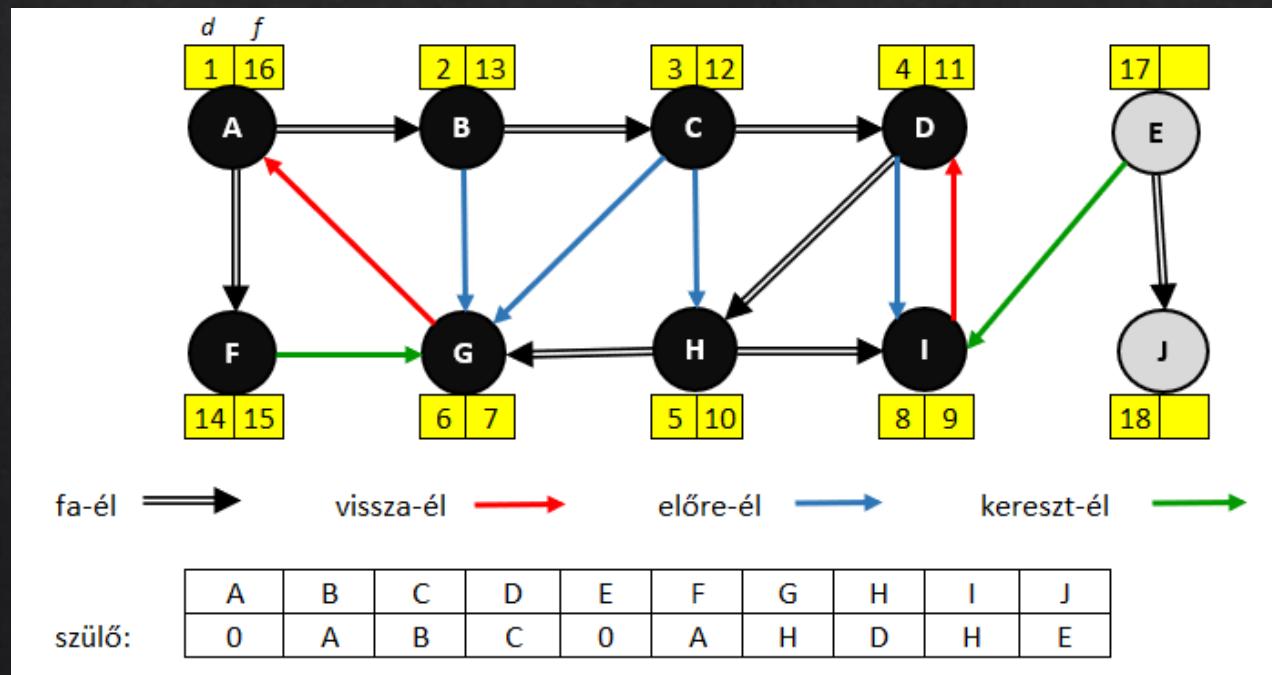
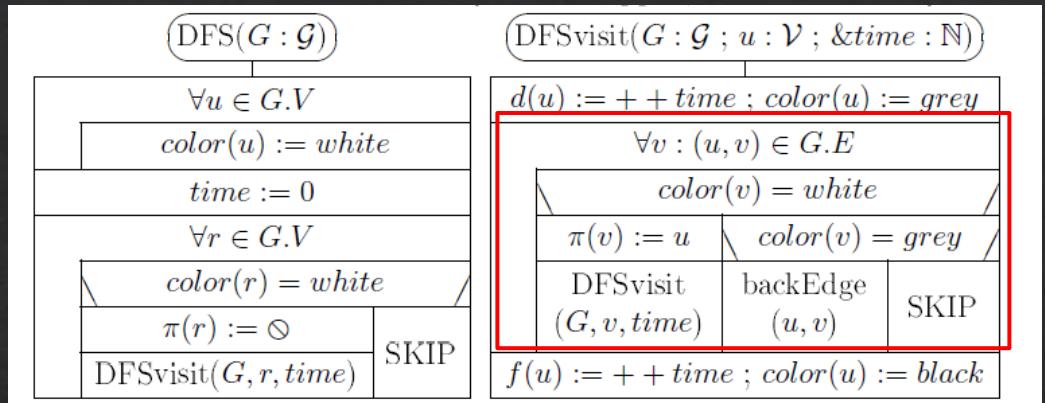
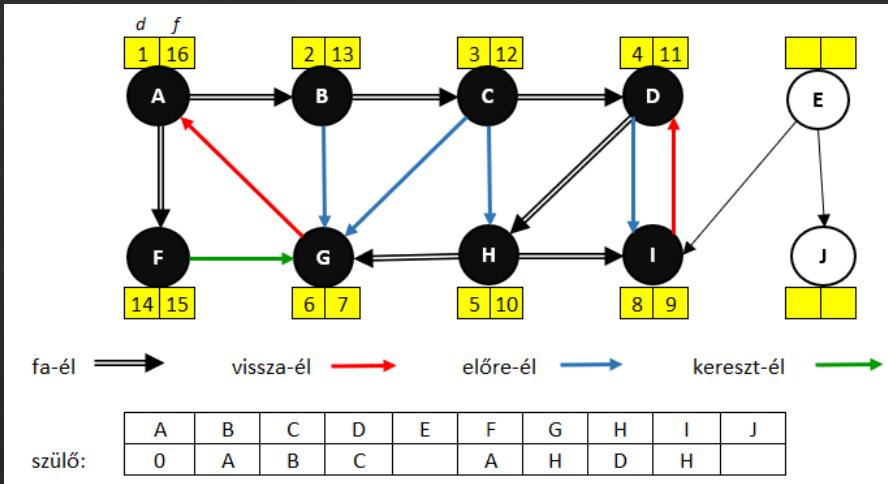
- ◆ A-ban vagyunk ismét, de van még fehér szomszédja: F
- ◆ „Kimegy” F-be
- ◆ $d(F) := 14$
- ◆ $\text{color}(F) := \text{grey}$



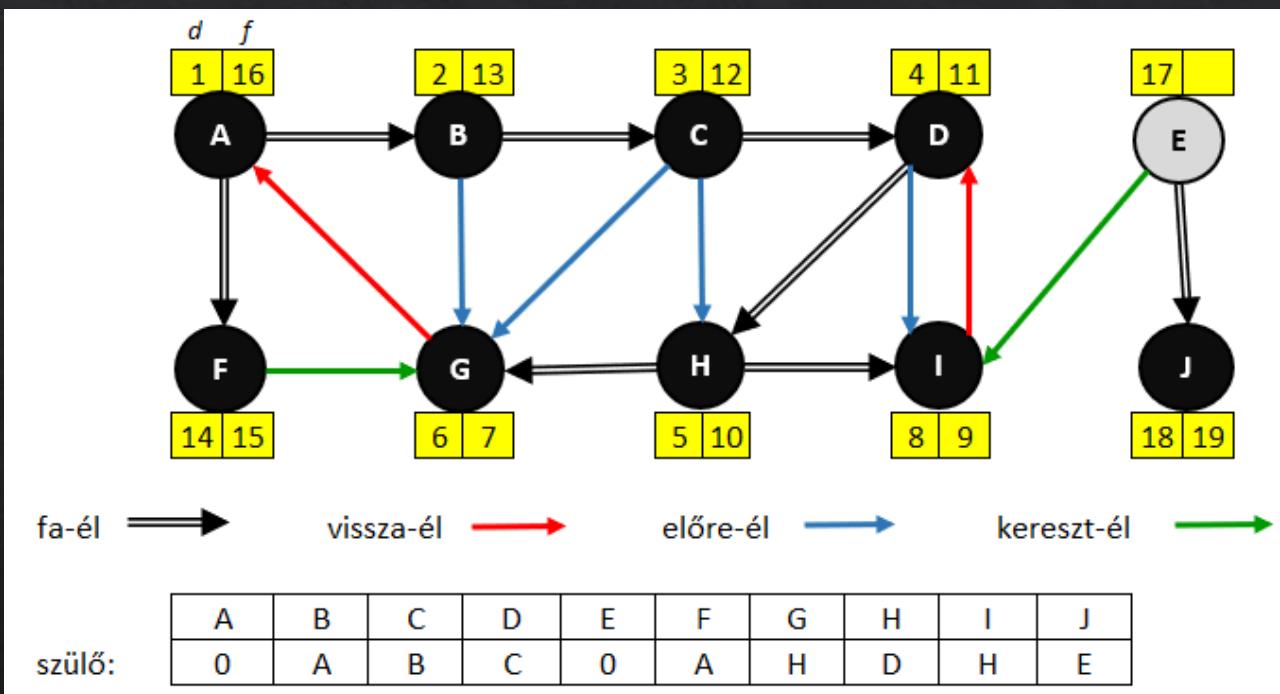
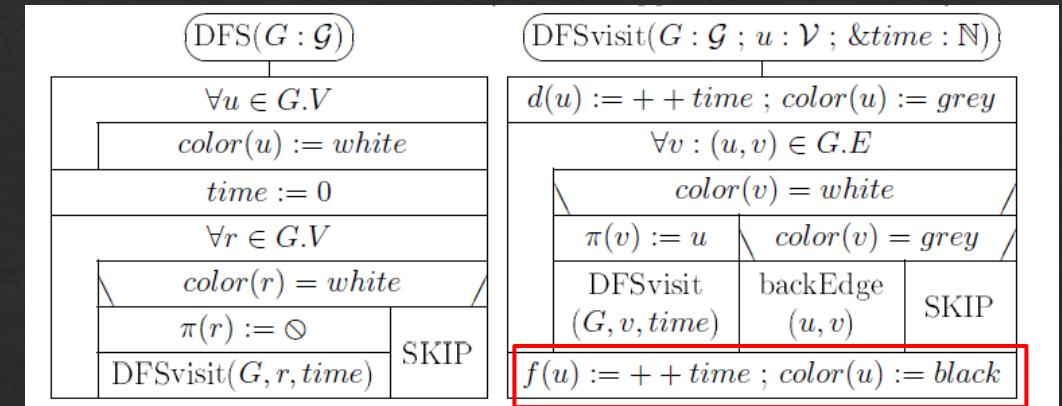
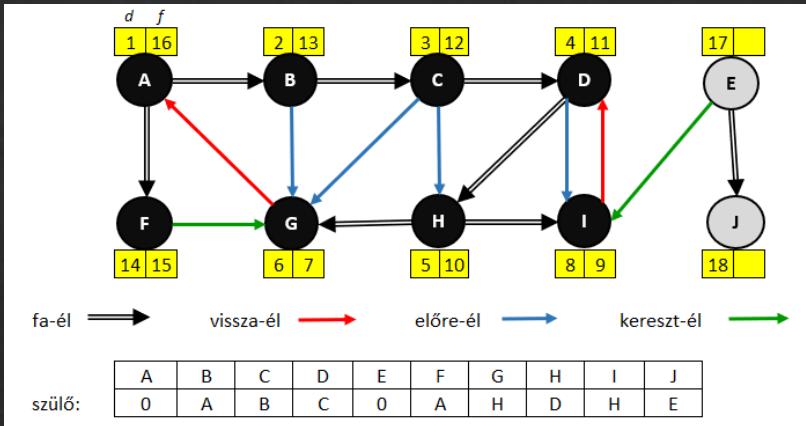
- ❖ F feldolgozása: G fekete szomszéd, $d(F) > d(G)$, tehát kereszt-élt találtunk.
- ❖ Befejeződik F feldolgozása
- ❖ Visszatér A csúcsba



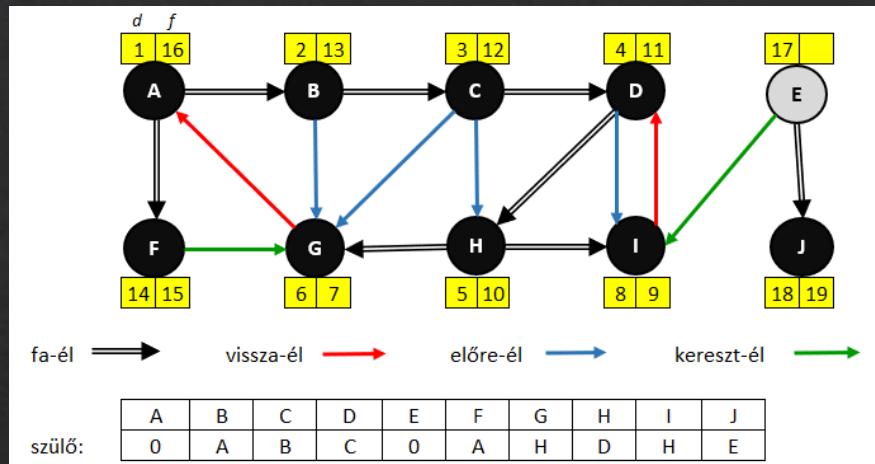
- ❖ A is kész csúcs lett
- ❖ Most tér vissza DFS algoritmusba, és vizsgálja, van-e még fehér csúcs
- ❖ Ha ábécé szerint megyünk, E következik.



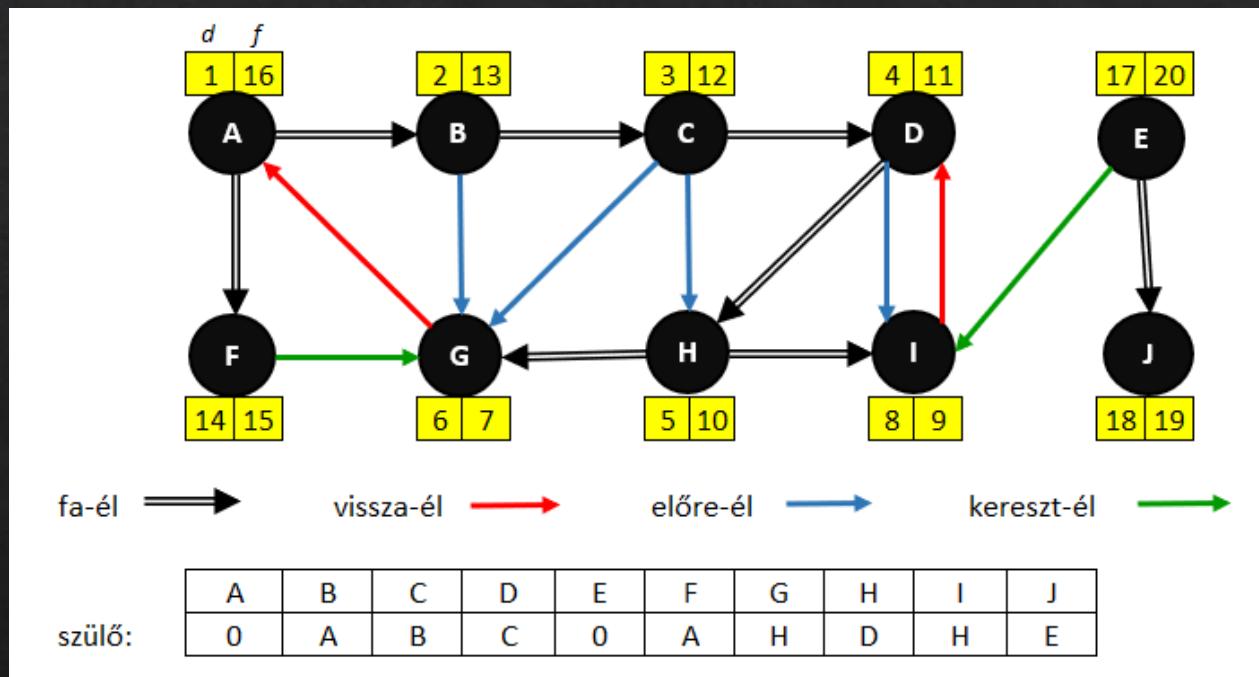
- ❖ E szomszédainak feldolgozása:
- ❖ I fekete, kereszt-él
- ❖ J fehér, fa-él



- ❖ J kész csúcs lett,
- ❖ Visszatér E-be

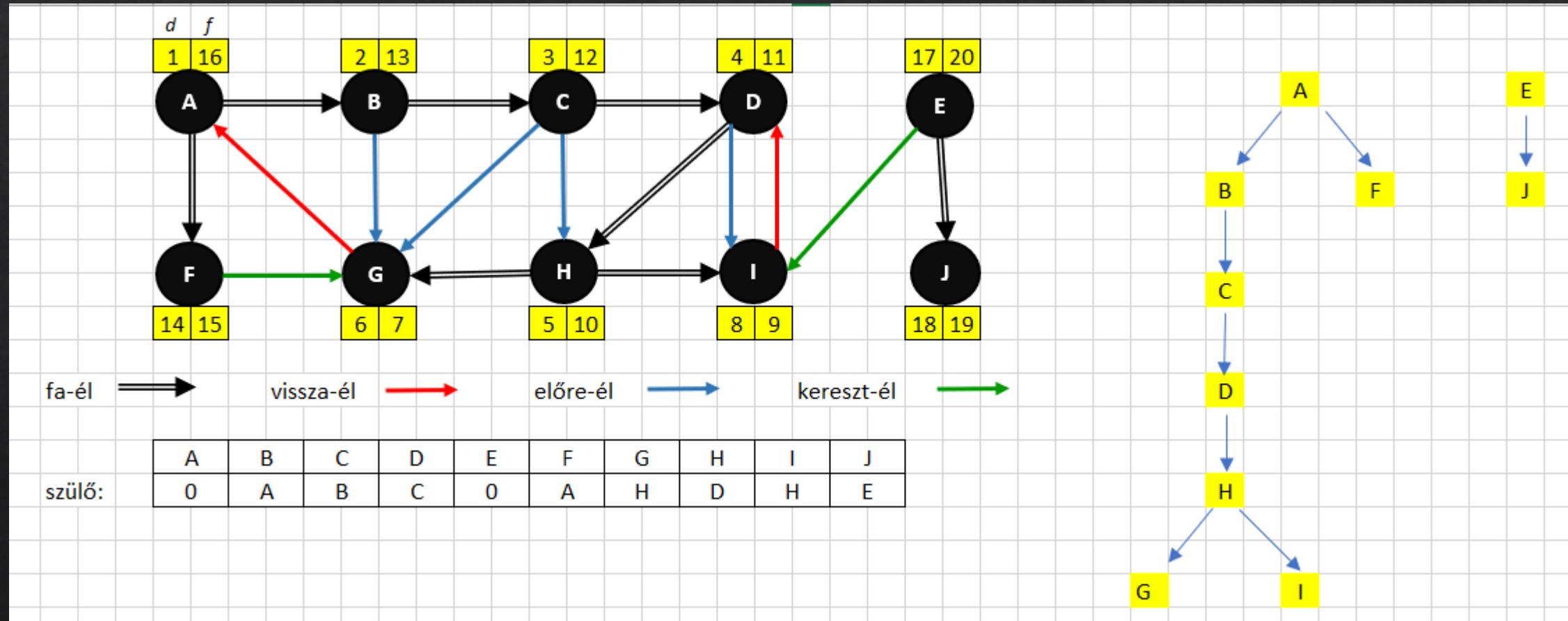


$\text{DFS}(G : \mathcal{G})$	$\text{DFSvisit}(G : \mathcal{G} ; u : \mathcal{V} ; \&time : \mathbb{N})$
$\forall u \in G.V$	$d(u) := + + time ; color(u) := \text{grey}$
$color(u) := \text{white}$	$\forall v : (u, v) \in G.E$
$time := 0$	$color(v) = \text{white}$
$\forall r \in G.V$	$\pi(v) := u$
\backslash	\backslash
$color(r) = \text{white}$	$color(v) = \text{grey}$
$\pi(r) := \oslash$	$\text{DFSvisit}(G, v, time)$
$\text{DFSvisit}(G, r, time)$	SKIP



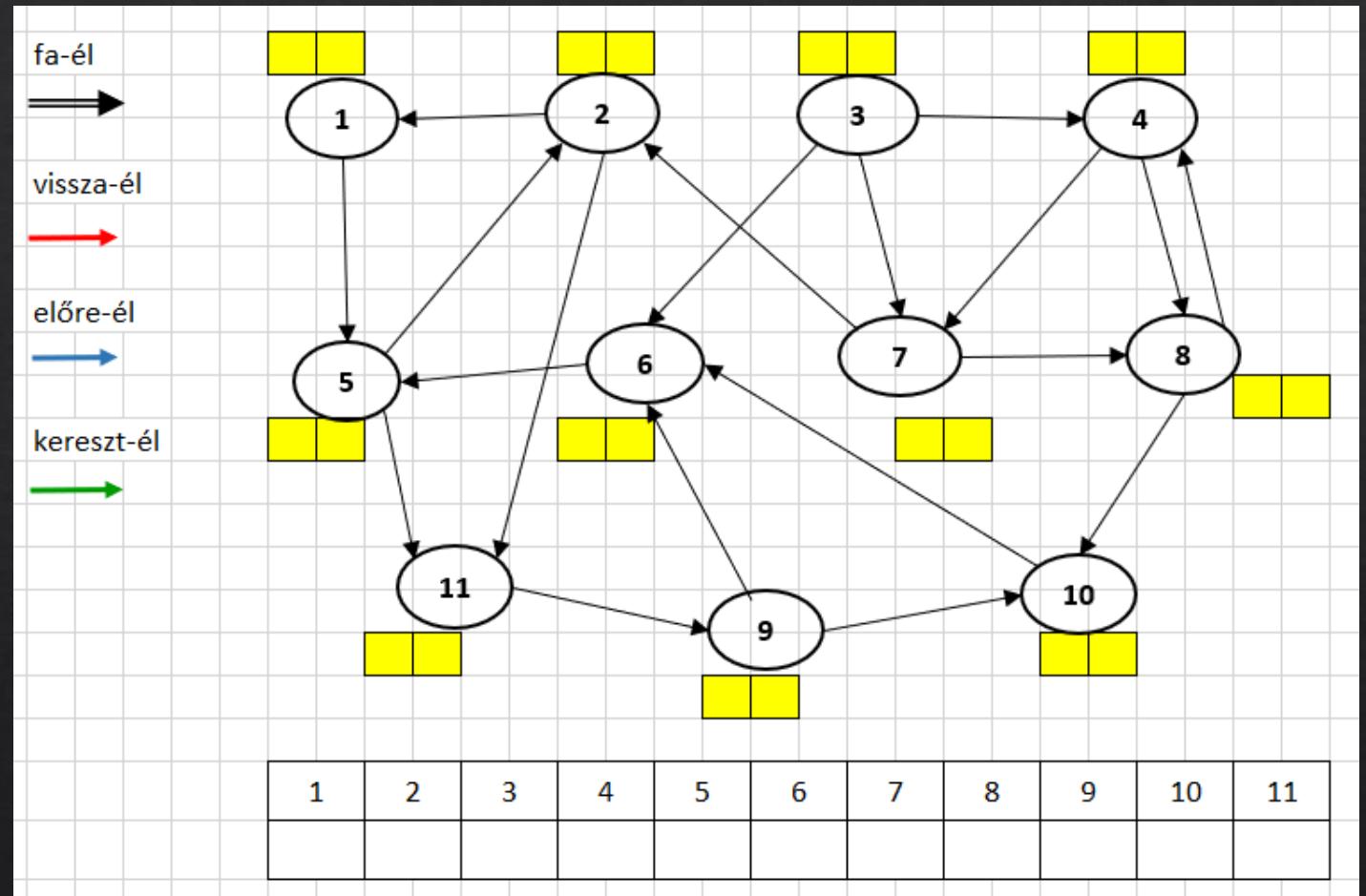
- ❖ E is kész csúcs,
- ❖ Nincs több fehér csúcsa a gráfnak,
- ❖ A bejárás véget ér.

- ◊ A kapott mélységi erdő
- ◊ A fák segítenek az élek ellenőrzésében!

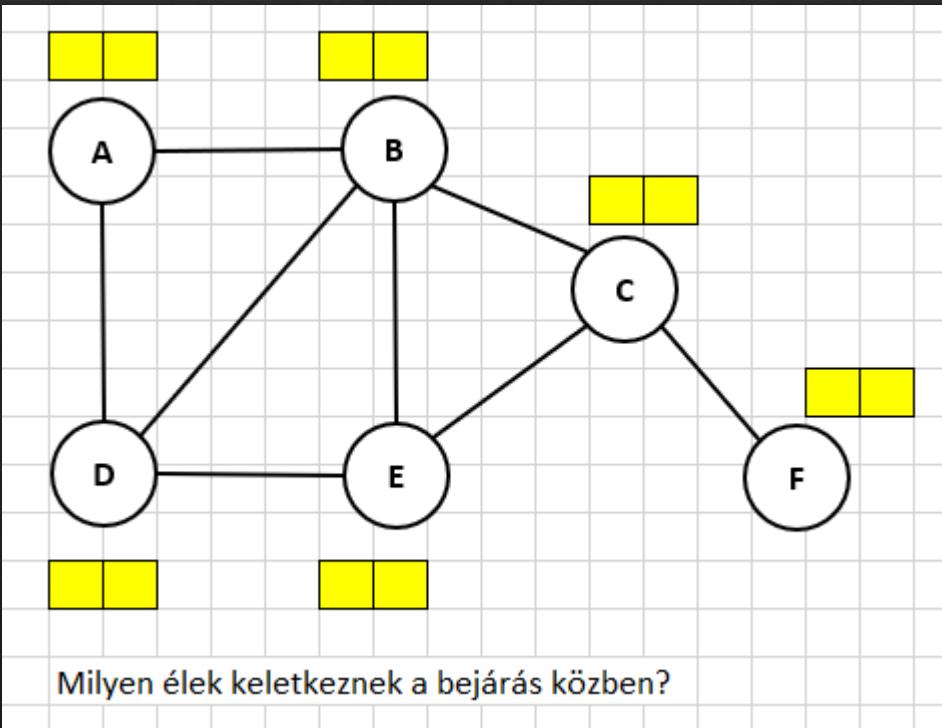


- ◆ Az előző példához hasonlóan mutassuk be a mélységi bejárás algoritmusát a mellékelt gráfon.
- ◆ Osztályozzuk a gráf éleit, rajzoljuk le a kapott mélységi erdőt.
- ◆ Segítség: melysegi_gyakorlo.xlsx

1. gyakorló feladat



Hogyan működik az algoritmus irányítatlan gráfon?



- ❖ Próbáljuk lefuttatni a példa gráfon.
- ❖ Figyeljük meg milyen típusúak lesznek az élek.
- ❖ Mivel itt egy él kétszer is megjelenik, akkor kap címkét, amikor elsőnek feltűnik.
- ❖ Segítség:
[melysegi_gyakorlo.xlsx](#)

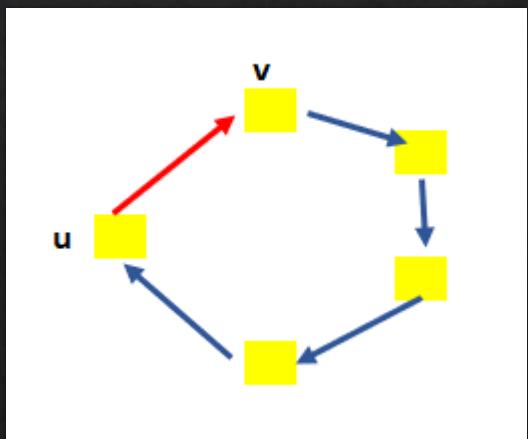
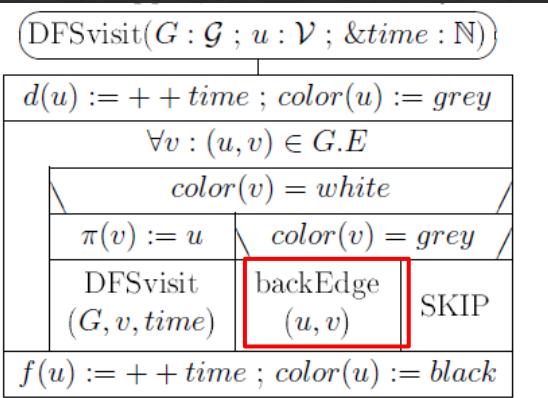
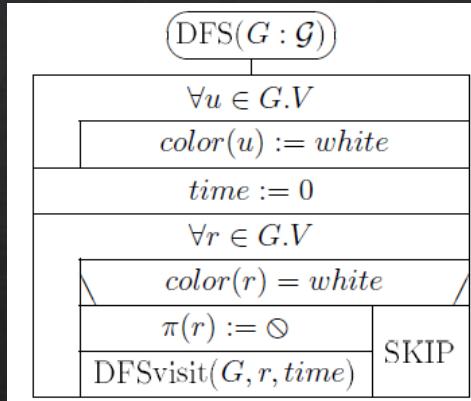
Implementálási kérdések

- ❖ Mi hagyható el?
- ❖ Szülő?
 - ❖ Igen, ha nem kell a mélységi fa.

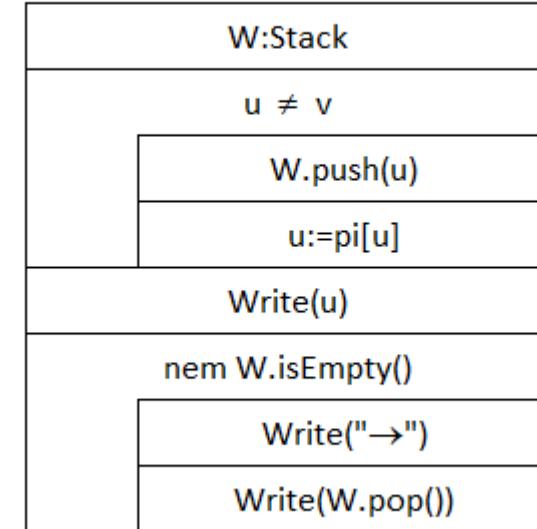
$\text{DFS}(G : \mathcal{G})$	$\text{DFSvisit}(G : \mathcal{G} ; u : \mathcal{V} ; \&time : \mathbb{N})$
$\forall u \in G.V$	$d(u) := + + time ; color(u) := \text{grey}$
$color(u) := \text{white}$	$\forall v : (u, v) \in G.E$
$time := 0$	$\backslash \quad color(v) = \text{white} \quad /$
$\forall r \in G.V$	$\pi(v) := u \quad \backslash \quad color(v) = \text{grey} \quad /$
$\backslash \quad color(r) = \text{white} \quad /$	$\text{DFSvisit}(G, v, time)$
$\pi(r) := \odot$	$\text{backEdge}(u, v)$
$\text{DFSvisit}(G, r, time)$	SKIP
	$f(u) := + + time ; color(u) := \text{black}$

- ❖ Színezés?
 - ❖ Igen, ha például az inicializáló ciklusban $d(u):=0$ és $f(u):=0$ értékkadások bekerülnek, akkor a szín helyettesíthető ezek vizsgálatával:
 $d(u) = 0$, akkor a csúcs fehér,
 $d(u) > 0$ és $f(u) = 0$, akkor a csúcs szürke,
 $d(u) > 0$ és $f(u) > 0$, akkor a csúcs fekete.
- ❖ Idő?
 - ❖ Elhagyható, persze akkor az éleket nem tudjuk osztályozni (miért?), de ha a színezést használjuk, a bejárás működik.
- ❖ time miért referencia paraméter?

Irányított kör kiírása



backEdge($\pi[1:N[n]], u, v: 1..n$)



Fejtörő kérdések

- ❖ Rajzoljon olyan három csúcsú, négy élű egyszerű gráfot, amelyben két egyszerű irányított kör van, és a DFS lehet, hogy megtalálja mindkettőt, de lehet, hogy csak az egyiket! Indokolja is az állítását!
- ❖ Adott egy irányított gráf, u és v a gráf két csúcsa. Létezik u és v között irányított út, $d(u) < d(v)$, és u mégsem őse v-nek a mélységi fában. Hogyan lehet ez?

Megfejtések a
következő gyakorlaton