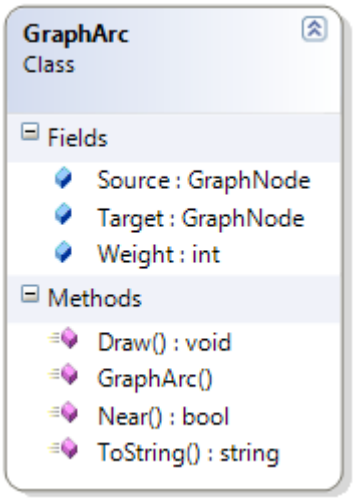
3. Beadandó feladat dokumentáció
Készítette: Koós Eszter EHA: KOEOACT.ELTE E-mail: eszterkoos@yahoo.com

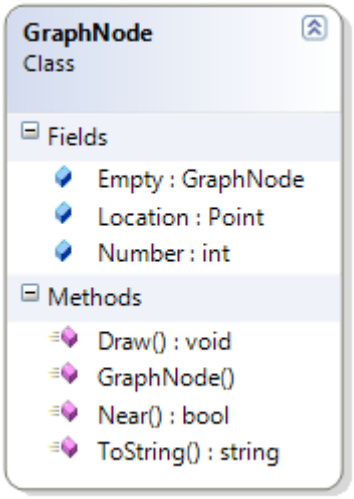
Feladat: Készítsünk programot, amely egy szöveges fájlban tárolt egyszerű irányított gráfot meg tud jeleníteni, és két megadott csúcs között megkeresi a legrövidebb utat. A szöveges input fájl első sorában a csúcsok n és az élek m száma található. Az ezt követő n sorban az egyes csúcsok sorszáma (1, 2, ..., n) és x, y koordinátái szerepelnek. Ezután m sorban az irányított élek vannak megadva a kezdőpontjukkal, végpontjukkal és nemnegatív költségükkel. A programban legyen lehetőség ilyen fájlok megnyitására (ellenőrizzük a formátumot). Ezután rajzolja ki a gráfot (jelenítse meg az élek költségét is), és ha a felhasználó rákattint egymás után két csúcsra, akkor keressen az elsőből a másodikba egy legrövidebb utat, és azt jelölje meg a gráfon. Ha újabb csúcspárra kattintunk, akkor az azok közötti legrövidebb út legyen kiemelve. Ha nincs irányított út a megadott két pont között, akkor azt írja ki a program. A legrövidebb utak keresésére a Dijkstra-algoritmust használjuk.

Elemzés:
- A gráf betöltését/mentését/új gráf létrehozását/algoritmusok indítását menüből lehet elérni
- A mentéshez/betöltéshez a windows szabvány file open/save dialogust használjuk.
- A gráfot grafikus felületen jelenítjük meg, ahol a csúcsokat és az éleket kirajzoljuk. A rajzolást egy panelen végezzük el, a panel alá egy listbox kerül, amin keresztül a program kommunikál a felhasználóval.

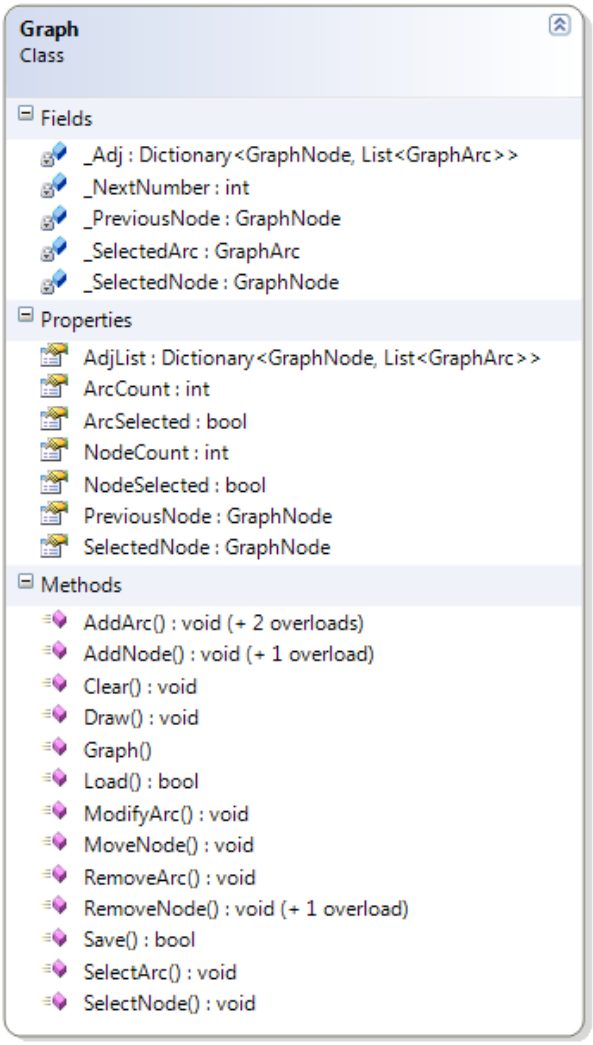
Rendszerterv:
A program két szintű: logikai és megjelenítési rétegekre tagozódik.
Logikai réteg osztályai



A GraphArc egy élet ír le két csúcs között.
Fieldjei:
Source – az irányított él talppontja
Target – az irányított él csúcsa
Weight – az irányított él súlya
Metódusai:
Draw – kirajzolja az élet megfelelő helyre és irányba nyíl alakban, paraméterei: Graphics gr – erre a graphicsra kell rajzolni, Color color – ezzel a színnel kell rajzolni, Font font – ezt a betűt kell használni kiíratáshoz
GraphArc – Konstruktor, paramétrei: GraphNode source – az él talpa, GraphNode target – az él csúcsa, Int32 weight – az él súlya.
Near – A paraméterben kapott pont az él közelében van, paraméterei: Point point
ToString – Az osztály szöveges reprezentációja



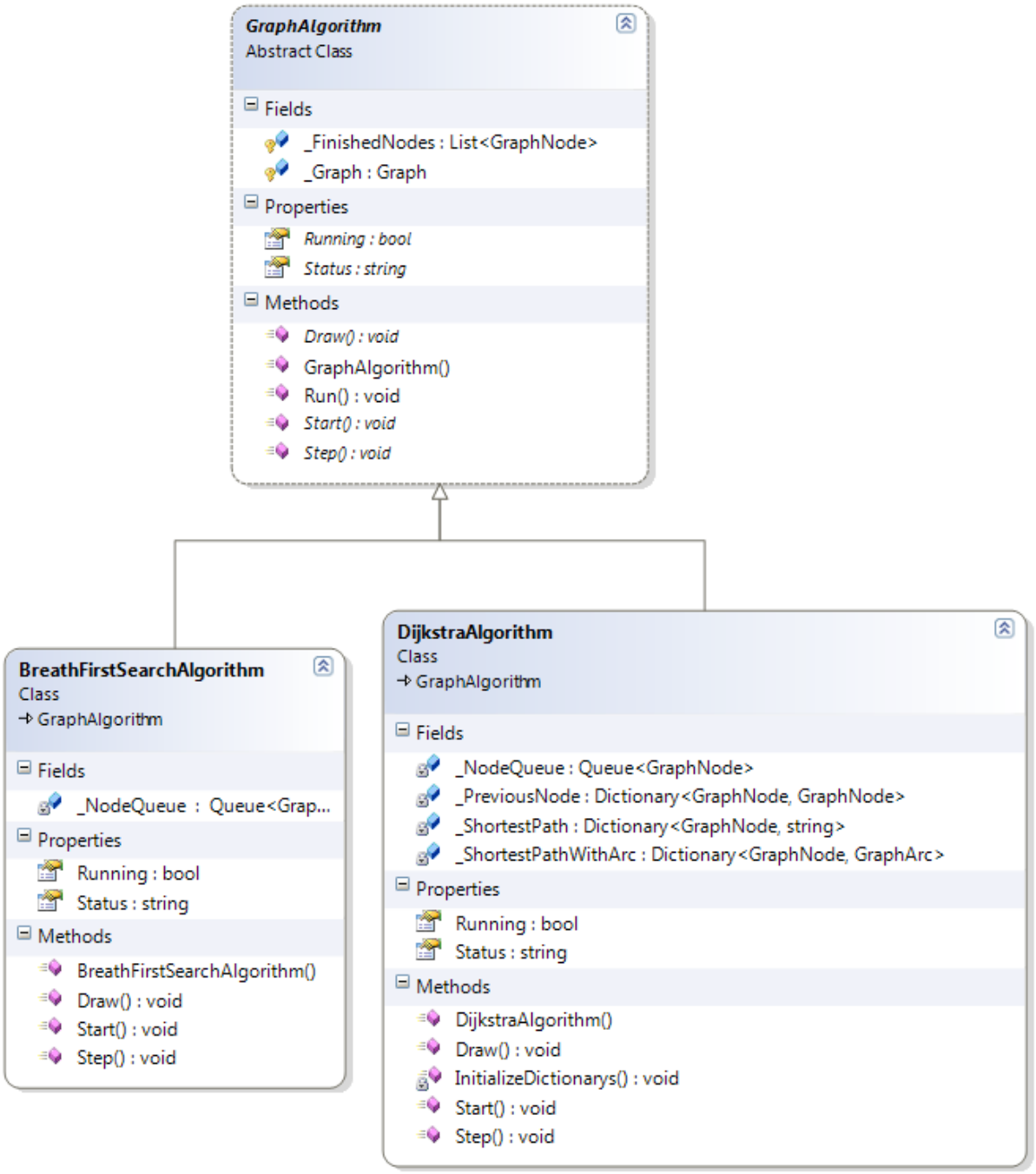
A GraphNode egy csúcsot ír let.
Fieldjei:
Empty – üres node, statikus
Location – a node helye
Number – a node neve
Metódusai:
Draw – kirajzolja a csúcsot a megfelelő helyre, paraméterei: Graphics gr – erre a graphicsra kell rajzolni, Color color – ezzel a színnel kell rajzolni, Font font – ezt a betűt kell használni kiíratáshoz
GraphNode – Konstruktor, paramétrei: int number, Point location.
Near – A paraméterben kapott pont az csúcs közelében van, paraméterei: Point point
ToString – Az osztály szöveges reprezentációja



A Graph egy gráfot ír le
Fieldjei:
_Adj – Egy csúcsból a szomszédaiba induló élek.
_NextNumber – a következő szabad szám a csúcsok elnevezéséhez
_PreviousNode – a jelenlegi kijelölés előtt kiválasztott csúcs.
_SelectedArc – jelenleg kiválasztott él
_SelectedNode – jelenleg kiválasztott csúcs
Propertik:
AdjList – Egy csúcsból a szomszédaiba induló élek listája az összes csúcsra.
ArcCount – élek száma
ArcSelected – van él kiválasztva
NodeCount – csúcsok száma
NodeSelected – van csúcs kiválasztva
PreviousNode – a jelenlegi kijelölés előtt kiválasztott csúcs.
SelectedNode – jelenleg kiválasztott csúcs
Metódusok:
AddArc – overloaded:
Point fromLocation, Point toLocation, Int32 weight – két pont közé próbál élt adni, ha a pontok csúcsra esnek.
Int32 fromNumber, Int32 toNumber, Int32 weight – két sorszámmal megadott csúcs közé ad élet.
GraphNode fromKey, GraphNode toKey, Int32 weight – megadott két csúcs közé tesz élet.
AddNode – overloaded:
Point location – az adott pontba elhelyez egy csúcsot
Int32 number, Point location – az adott pontba elhelyez egy csúcsot adott névvel

Clear – Törli a gráfot

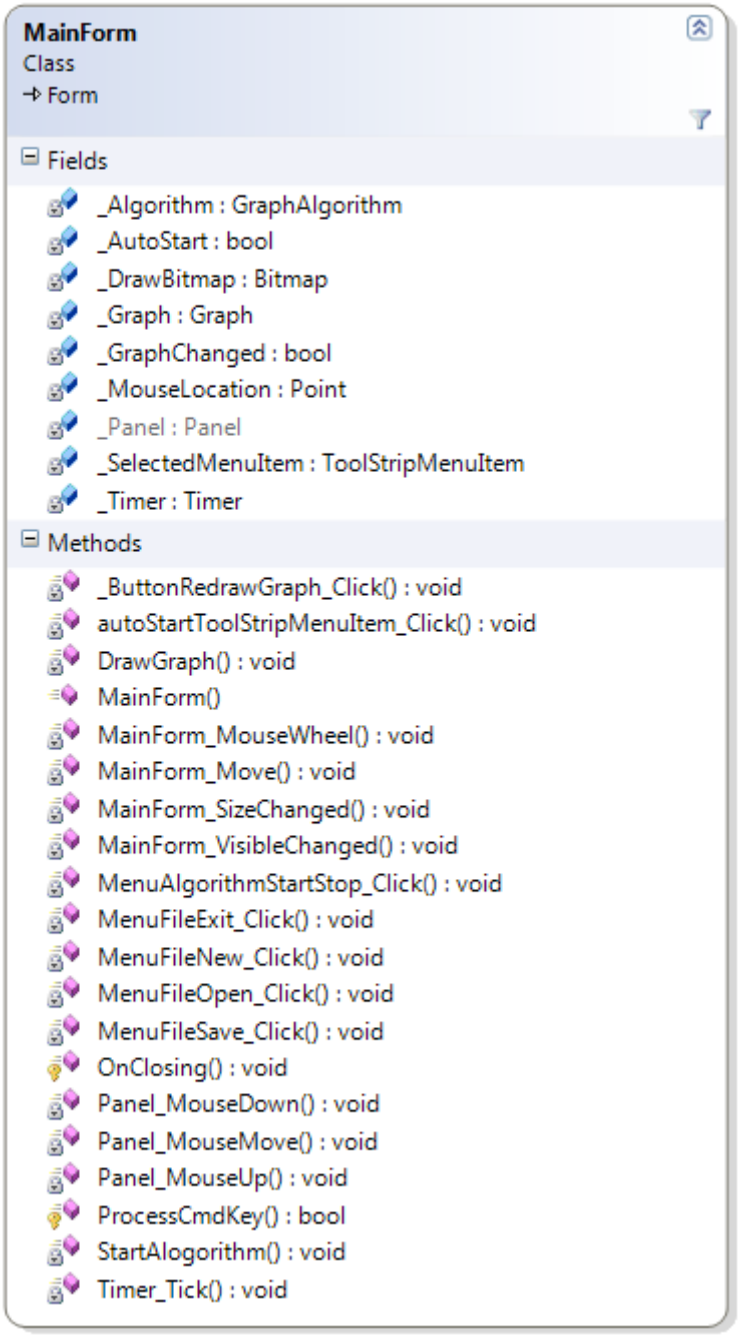
Draw – kirajzolja a gráfot
ModifyArc – paraméter: Int32 weightModifier – a kiválasztott él súlyát módosítja.
MoveNode – paraméter: Point targetLocation – az adott pontba áthelyezi a csúcsot
RemoveArc – letörli a kiválasztott csúcsot
RemoveNode – overloaded:
Paraméter nélkül törli a kiválasztott csúcsot, az oda érkező és belőle induló éleket.
GraphNode removeKey – törli a paraméterként kapott csúcsot, az oda érkező és belőle induló éleket.
Save – Menti a gráfot
SelectArc – Az adott pontban lévő élet kiválasztja
SelectNode – Az adott pontban lévő csúcsot kiválasztja



A két megvalósított algoritmus a GraphAlgorithm absztrakt osztályból származik. A feladatra koncentrálva most csak a DijkstraAlgorithm leírására szorítkozunk.

Fieldjei:
_NodeQueue – tárolja a még fldolgozásra váró csúcsokat
_PreviousNode – tárolja a listát, hogy egy adott csúcsba mely élből lehet a legolcsóbban eljutni
_ShortestPath – lista olyan értékpárokkal, amelyek az egyes nodeokhoz tartozó költésget mutatják.
_ShortestParhWithArc – az előző kettő vegyítése, olyan kulcs-érték lista, melyben az adott csúcsba vezető legolcsóbb él van tárolva.
Propertik:
Running – Fut az algoritmus
Status – az algoritmus jelenlegi állapota
Metódusok:
DijkstraAlgorithm – Kontruktor, paramétere egy Graphics, amire rajzolni kell.
Draw – paramétere egy graphics, amire kirajzolja a jelenlegi elemzési állapotot, pirossal a jelenleg elemzett részeket, zölddel az eddigi legrövidebb utat, kékkel a már elemzett, de nem a legrövidebb utat tartalmazó részeket, feketével az el nem érhető részeket.
InitializeDictionarys – feltölti az alapértékekkel a távolságokat tartalmazó Dictionaryt
Start – inicializálja az algoritmust
Step – lépteti az algoritmust

A grafikus felület a MainForm osztályban van megvalósítva. A controlok fieljeit a felsorolásból az egyszerűség kedvéért kihagytuk.



Fieldjei:
_Algorithm – a kiválasztott algoritmust tartalmazza
_AutoStart – amennyiben a Dijkstra algoritmust menüben az AutoStart ki an választva ez a filed true. Ez azt jelenti, hog minden esetben, amikor egymás után két csúcsot kiválasztunk, a program automatikusan megpróbálja megkeresni a köztük lévő legrövidebb utat.
_DrawBitmap – az a bitmap, amire a gráf valójában rajzolva lesz.
_Graph – a gráf osztály példánya, amin dolgozunk.
_GraphChanged – módosítást hajtottak végre a gráfon
_MousePoint – Csúcs dragdropnál tartalmazza a célkoordinátákat.
_Panel – ez a tényleges megjelenítő felület, a _DrawBitmap itt van megjelenítve
_SelecedMenuItem – Meghatározza használandó algoritmust.
Metódusok:
_ButtonRedrawGraph_Click – elemzés végén nincs automatikusan újrarajzolva a gráf, hogy ki lehessen értékelni a futtatás eredményét. Ez gomb indítja az újrarajzolást.

autoStartToolStripMenuItem_Click – beállítja az _AutoStart fieldet.
DrawGraph – felhívja a gráf vagy az algoritmus rajzoló rutinját
MainForm – konstruktor
MainForm_MouseWheel – módosítja az él értékét
MainForm_Move – újrarajzolja a gráfot
MainForm_SizeChanged – újrarajzolja a gráfot
MainForm_VisibleChanged – újrarajzolja a gráfot
MenuAlgorithmStartStop_Click – indítja vagy megállítja a megfelelő algoritmust
MenuFileExit_Click – tájékoztatás uzán kilép
MenuFileNew_Click – tájékoztatás után törli a jelenlegi gráfot
MenuFileOpen_Click – tájékoztatás után megnyit egy előzőleg mentett gráfot.
MenuFileSave_Click – elmenti a gráfot
OnClosing – tájékoztatás után megszakítja az alkalmazás bezárását, ha a felhasználó úgy dönt
Panel_MouseDown – kiválaszt egy csúcsot vagy élet, illetve elkezdi egy új él rajzolását
Panel_MouseMove – ha új él rajzolása inicializálva van, rajzol egy új élet.
Panel_MouseUp – ha él rajzolása folyamatban a gráfon keresztül megpróbálja létrehozni az új élet.
ProcessCmdKey – kezeli a törlés, fel/le gombkat.
Timer_Tick – ütemező metódus az algoritmus végrehajtásához

Implementáció:
- A főprogram feladata a grafikus felület betöltése.
- A MainForm biztosítja a felhasználó interakciót illetve a megjelenítést.
- Az algoritmus:
Minden lépésben tartsuk nyilván az összes csúcsra, a forrástól az illető csúcsba vezető, eddig talált legrövidebb utat (a már megismert módon a d[1..n] tömbben a távolságot, és P[1..n] tömbben a megelőző csúcsot).
1) Kezdetben a távolság legyen a kezdőcsúcsra 0, a többi csúcsra ∞ .
2) Minden lépésben a nem KÉSZ csúcsok közül tekintsük az egyik legkisebb távolságú (min d) csúcsot:
a) Azt mondhatjuk, hogy ez a $v \in V$ csúcs már KÉSZ, azaz ismert a hozzá vezető legrövidebb út.
b) A v-t terjesszük ki, azaz v csúcs szomszédaira számítsuk ki a (már ismert) v-be vezető, és onnan egy kimenő éllel meghosszabbított út hosszát. Amennyiben ez jobb (kisebb), mint az illető szomszédba eddig talált legrövidebb út, akkor innentől kezdve ezt az utat tekintsük, az adott szomszédba vezető, eddig talált legrövidebb útnak. Ezt az eljárást szokás közelítésnek is nevezni.

Tesztelés:
- egyszerű gráfon az útvonal tervezése, ellenőrzés manuálisan
- összetett, köröket tertalmazó gráfon az útvonal tervezése, ellenőrzés manuálisan
- el nem érhető csúcsokhoz útvonal tervezése
- automatikus gyorstervezés teszetelése