

NHF – Vonatjegy  
Programozás alapjai 2.  
DORAU5

Toldi Lőrinc László  
2024.05.19.

## Feladat

Tervezze meg egy vonatjegy eladó rendszer egyszerűsített objektummodelljét, majd valósítsa azt meg! A vonatjegy a feladatban mindig jegyet és helyjegyet jelent együtt. Így egy jegyen minimum a következőket kell feltüntetni:

- vonatszám, kocsiszám, hely
- indulási állomás, indulási idő
- érkezési állomás, érkezési idő

A rendszerrel minimum a következő műveleteket kívánjuk elvégezni:

- vonatok felvétele
- jegy kiadása

A rendszer később lehet bővebb funkcionalitású (pl. késések kezelése, vonat törlése, menetrend, stb.), ezért nagyon fontos, hogy jól határozza meg az objektumokat és azok felelősségét.

Valósítsa meg a jeggyel végezhető összes értelmes műveletet operátor átdefiniálással (overload), de nem kell ragaszkodni az összes operátor átdefiniálásához!

## Terv

### A program célja

A feladat egy vonatjegy eladó rendszer megvalósítása, amely képes kezelni a vonatok rendszerbe történő felvételét és törlését, illetve a már mentett vonatokra szóló jegyek kiadását. A rendszerben regisztrált vonatok listájából menetrend lekérdezhető. A program képes a vonatokat és kiadott jegyeket fájl-ba kiírni és fájl-ból beolvasni. A program menüvezérelt, grafikus megjelenítést nem alkalmaz.

### menü megjelenése

A menü a konzolban jelenik meg szövegesen. A menüpontok egymást követő sorokba vannak írva, minden sor a megfelelő sorszámmal kezdődik. A választható menüpontok a következők; (1) jegykiadás, (2) vonat felvétel, (3) vonat törlés, (4) adatok mentése, (5) adatok betöltése, (6) menetrend lekérdezés, (7) kilépés.

### menüpont választás

A menüpont választás az ahhoz tartozó szám beírásával majd enter-el történik. (pl. (1) - Jegykiadás esetében az '1' konzolban való bevitelével.) Hibás bevitelével esetén a program a felhasználót tájékoztatja és újbóli választásra kéri. A menüpont választás után a választott menüpont jelenik meg.

### jegykiadás

A jegykiadás menüpontban egy jegy kiadására van lehetőség. A felhasználónak elsőnek az indulási és az érkezési állomás nevét kell megadnia, majd az indulási időpontot. A program arra a vonatra ad jegyet, mely az adott állomásra ahhoz leghamarabb érkezik. Amennyiben nem találtunk vonatot a kettő állomás között a felhasználót tájékoztatjuk és felajánljuk a folyamat újrakezdését. Ha találtunk vonatot az indulás időpontját kiírjuk. A felhasználónak a jegy típusát is kikell választania, amelynek

ára a menetidőből ered (x Ft/perc alapon). A jegyek típusai a következők: első osztály - 48Ft/perc, második osztály - 30Ft/perc, diák 12Ft/perc, nyugdíjas 12Ft/perc. Az adatok bekérése egyesével és egymást követően történik. A készített objektumot a rendszerben tároljuk. A 'kiadott' jegy a konzolban jelenik meg.

### **vonat felvétel**

A vonat felvétel menüpontban új vonat regisztrálására van lehetőség. A felhasználótól a következő adatokat kérjük be; vonatszám [egész], kocsi szám [egész], kocsikban elérhető helyek száma [egész], állomások száma [egész, minimum 2]. A megadott állomások számának megfelelően annyszor jelenik meg az állomás bekérése, melyek sorrendje egymást követő kell, hogy legyen. Az állomás bekért attribútumai: állomás neve [szöveg], érkezési idő [óra: egész, perc: egész]. A készített objektumot a rendszerben tároljuk. Hiba esetén a felhasználót tájékoztatjuk és újbóli bevitelre kérjük. Sikeres esetben visszakerülünk a főmenübe.

### **vonat törlés**

A vonat törlése menüpontban létező vonat törlése elérhető. A felhasználótól a vonat számát kérjük be, majd az ahhoz tartozó vonatot töröljük a rendszerből. Hiba esetén a felhasználót tájékoztatjuk és újbóli bevitelre kérjük. Sikeres esetben visszakerülünk a főmenübe.

### **adatok mentése**

Az adatok mentése menüpontban fájl-ba tudjuk kiírni a vonatokat és a hozzájuk tartozó állomásokat, kocsikat és jegyeket. A felhasználótól a fájl nevét kérjük be. Hiba esetén a felhasználót tájékoztatjuk és újbóli bevitelre kérjük. Sikeres esetben visszakerülünk a főmenübe.

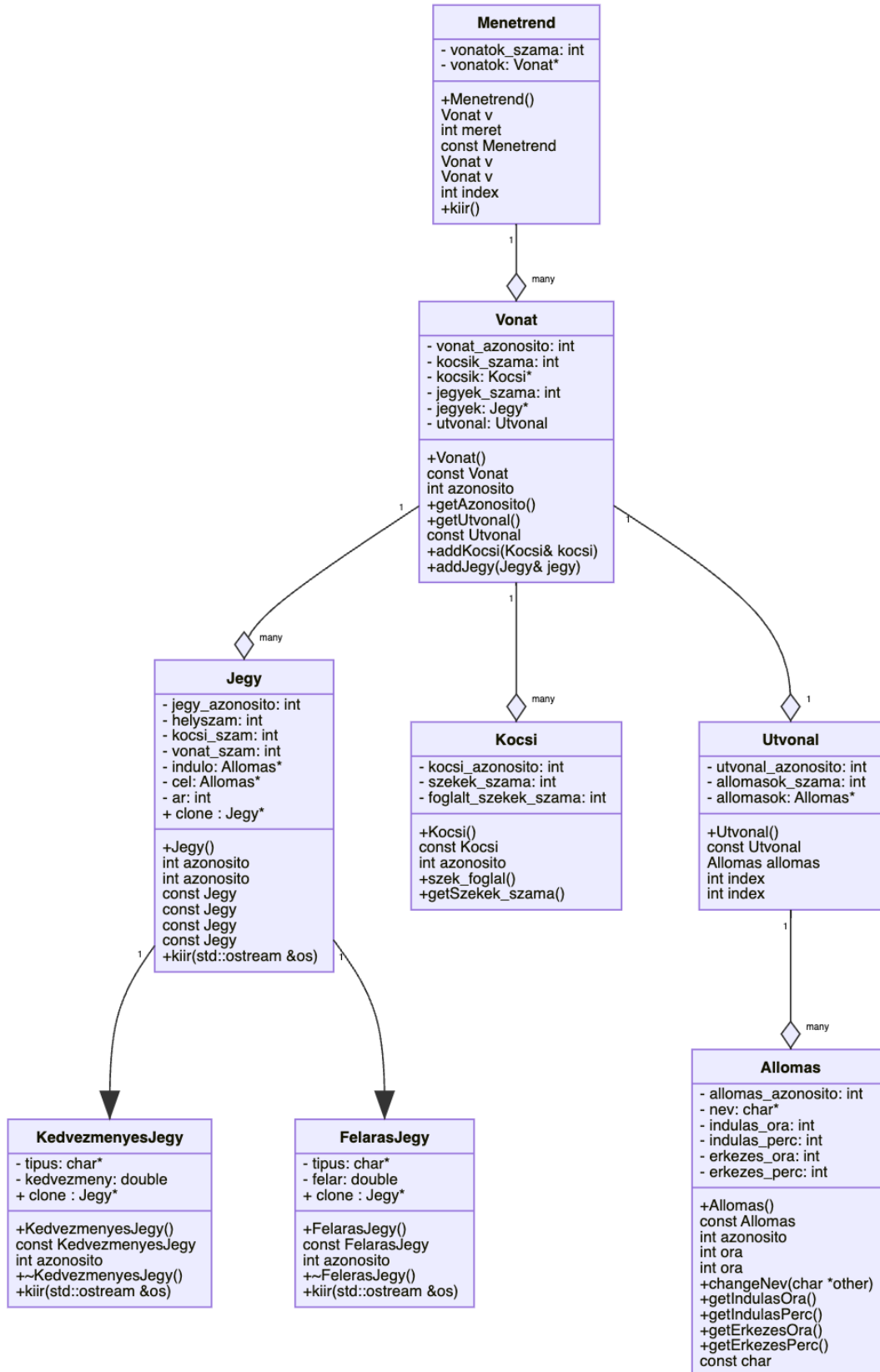
### **adatok betöltése**

Az adatok betöltése menüpontban fájl-ból olvassuk be a vonatokat és a hozzájuk tartozó állomásokat, kocsikat és jegyeket. A beolvasott adatokat a rendszerben tároljuk. A felhasználótól a fájl nevét kérjük be. Hiba esetén a felhasználót tájékoztatjuk és újbóli bevitelre kérjük. Sikeres esetben visszakerülünk a főmenübe.

### **menetrend lekérdezés**

A menetrend lekérdezésnél a következő paramétereket kérjük be és azokat szűrőként alkalmazzuk a menetrendre; Indulási állomás ["szöveg"], érkezési állomás ["szöveg"]. Az adatok bekérése egyesével és egymást követően történik. Az üresen hagyott paramétereket nem alkalmazzuk. Az adatok bevitelét követően kiírjuk a megfelelő vonatok számát és listázzuk őket. kilépés A kilépés menüpont leállítja a programot.

## Objektum terv



# NHF Dokumentáció (Doxygen)



|  |          |
|--|----------|
| <b>1. Hierarchikus mutató</b>                            | <b>1</b> |
| 1.1. Osztályhierarchia .....                             | 1        |
| <b>2. Osztálymutató</b>                                  | <b>3</b> |
| 2.1. Osztálylista .....                                  | 3        |
| <b>3. Fájlmutató</b>                                     | <b>5</b> |
| 3.1. Fájllista .....                                     | 5        |
| <b>4. Osztályok dokumentációja</b>                       | <b>7</b> |
| 4.1. Allomas osztályreferencia.....                      | 7        |
| 4.1.1. Részletes leírás .....                            | 8        |
| 4.1.2. Konstruktork és destruktorok dokumentációja ..... | 8        |
| 4.1.2.1. Allomas() [1/3] .....                           | 8        |
| 4.1.2.2. Allomas() [2/3] .....                           | 8        |
| 4.1.2.3. Allomas() [3/3] .....                           | 8        |
| 4.1.2.4. ~Allomas() .....                                | 10       |
| 4.1.3. Tagfüggvények dokumentációja .....                | 10       |
| 4.1.3.1. changeErkezes() .....                           | 10       |
| 4.1.3.2. changeIndulas().....                            | 10       |
| 4.1.3.3. changeNev() .....                               | 11       |
| 4.1.3.4. getErkezes().....                               | 11       |
| 4.1.3.5. getErkezesOra() .....                           | 11       |
| 4.1.3.6. getErkezesPerc().....                           | 11       |
| 4.1.3.7. getIndulas() .....                              | 11       |
| 4.1.3.8. getIndulasOra().....                            | 12       |
| 4.1.3.9. getIndulasPerc() .....                          | 12       |
| 4.1.3.10. getNev().....                                  | 12       |
| 4.1.3.11. operator=() .....                              | 12       |
| 4.1.3.12. read().....                                    | 12       |
| 4.1.3.13. write().....                                   | 13       |
| 4.1.4. Adattagok dokumentációja.....                     | 13       |
| 4.1.4.1. allomas_azonosito .....                         | 13       |
| 4.1.4.2. erkezes .....                                   | 13       |
| 4.1.4.3. indulas .....                                   | 13       |
| 4.1.4.4. nev.....  | 13       |
| 4.2. FelarasJegy osztályreferencia .....                 | 14       |
| 4.2.1. Részletes leírás .....                            | 15       |
| 4.2.2. Konstruktork és destruktorok dokumentációja ..... | 15       |
| 4.2.2.1. FelarasJegy() [1/3] .....                       | 15       |
| 4.2.2.2. FelarasJegy() [2/3] .....                       | 15       |
| 4.2.2.3. FelarasJegy() [3/3] .....                       | 16       |
| 4.2.2.4. ~FelarasJegy().....                             | 16       |

|           |  |    |
|-----------|--|----|
| 4.2.3.    | Tagfüggvények dokumentációja                 | 16 |
| 4.2.3.1.  | clone()                                      | 16 |
| 4.2.3.2.  | kiir()                                       | 17 |
| 4.2.3.3.  | operator=()                                  | 17 |
| 4.2.3.4.  | read()                                       | 17 |
| 4.2.3.5.  | write()                                      | 17 |
| 4.2.4.    | Adattagok dokumentációja                     | 18 |
| 4.2.4.1.  | felar  | 18 |
| 4.2.4.2.  | tipus  | 18 |
| 4.3.      | Ido osztályreferencia                        | 18 |
| 4.3.1.    | Részletes leírás                             | 19 |
| 4.3.2.    | Konstruktorok és destruktorok dokumentációja | 19 |
| 4.3.2.1.  | Ido() [1/3]                                  | 19 |
| 4.3.2.2.  | Ido() [2/3]                                  | 19 |
| 4.3.2.3.  | Ido() [3/3]                                  | 19 |
| 4.3.2.4.  | ~Ido()                                       | 20 |
| 4.3.3.    | Tagfüggvények dokumentációja                 | 20 |
| 4.3.3.1.  | addOra()                                     | 20 |
| 4.3.3.2.  | addPerc()                                    | 20 |
| 4.3.3.3.  | getOra()                                     | 20 |
| 4.3.3.4.  | getPerc()                                    | 20 |
| 4.3.3.5.  | kiir()                                       | 21 |
| 4.3.3.6.  | operator"!="()                               | 21 |
| 4.3.3.7.  | operator=="()                                | 21 |
| 4.3.3.8.  | read()                                       | 21 |
| 4.3.3.9.  | setIdo()                                     | 22 |
| 4.3.3.10. | setOra()                                     | 22 |
| 4.3.3.11. | setPerc()                                    | 22 |
| 4.3.3.12. | write()                                      | 22 |
| 4.3.4.    | Adattagok dokumentációja                     | 24 |
| 4.3.4.1.  | ora  | 24 |
| 4.3.4.2.  | perc   | 24 |
| 4.4.      | Jegy osztályreferencia                       | 24 |
| 4.4.1.    | Részletes leírás                             | 25 |
| 4.4.2.    | Konstruktorok és destruktorok dokumentációja | 26 |
| 4.4.2.1.  | Jegy() [1/4]                                 | 26 |
| 4.4.2.2.  | Jegy() [2/4]                                 | 26 |
| 4.4.2.3.  | Jegy() [3/4]                                 | 26 |
| 4.4.2.4.  | Jegy() [4/4]                                 | 27 |
| 4.4.2.5.  | ~Jegy()                                      | 27 |
| 4.4.3.    | Tagfüggvények dokumentációja                 | 27 |
| 4.4.3.1.  | clone()                                      | 27 |



|          |  |    |
|----------|--|----|
| 4.4.3.2. | getAr()                                      | 27 |
| 4.4.3.3. | kiir()                                       | 27 |
| 4.4.3.4. | operator"!=()                                | 28 |
| 4.4.3.5. | operator=()                                  | 28 |
| 4.4.3.6. | operator==()                                 | 28 |
| 4.4.3.7. | read()                                       | 29 |
| 4.4.3.8. | write()                                      | 29 |
| 4.4.4.   | Adattagok dokumentációja                     | 29 |
| 4.4.4.1. | ar   | 29 |
| 4.4.4.2. | cel_ido                                      | 29 |
| 4.4.4.3. | cel_nev                                      | 30 |
| 4.4.4.4. | helyszam                                     | 30 |
| 4.4.4.5. | indulo_ido                                   | 30 |
| 4.4.4.6. | indulo_nev                                   | 30 |
| 4.4.4.7. | jegy_azonosito                               | 30 |
| 4.4.4.8. | kocsi_szam                                   | 30 |
| 4.4.4.9. | vonat_szam                                   | 30 |
| 4.5.     | KedvezmenyesJegy osztályreferencia           | 31 |
| 4.5.1.   | Részletes leírás                             | 32 |
| 4.5.2.   | Konstruktorok és destruktorok dokumentációja | 32 |
| 4.5.2.1. | KedvezmenyesJegy() [1/3]                     | 32 |
| 4.5.2.2. | KedvezmenyesJegy() [2/3]                     | 32 |
| 4.5.2.3. | KedvezmenyesJegy() [3/3]                     | 33 |
| 4.5.2.4. | ~KedvezmenyesJegy()                          | 33 |
| 4.5.3.   | Tagfüggvények dokumentációja                 | 33 |
| 4.5.3.1. | clone()                                      | 33 |
| 4.5.3.2. | kiir()                                       | 34 |
| 4.5.3.3. | operator=()                                  | 34 |
| 4.5.3.4. | read()                                       | 34 |
| 4.5.3.5. | write()                                      | 34 |
| 4.5.4.   | Adattagok dokumentációja                     | 35 |
| 4.5.4.1. | kedvezmeny                                   | 35 |
| 4.5.4.2. | tipus  | 35 |
| 4.6.     | Kocsi osztályreferencia                      | 35 |
| 4.6.1.   | Részletes leírás                             | 36 |
| 4.6.2.   | Konstruktorok és destruktorok dokumentációja | 36 |
| 4.6.2.1. | Kocsi() [1/3]                                | 36 |
| 4.6.2.2. | Kocsi() [2/3]                                | 36 |
| 4.6.2.3. | Kocsi() [3/3]                                | 36 |
| 4.6.2.4. | ~Kocsi()                                     | 37 |
| 4.6.3.   | Tagfüggvények dokumentációja                 | 37 |
| 4.6.3.1. | getAzonosito()                               | 37 |

|           |   |    |
|-----------|---|----|
| 4.6.3.2.  | <a href="#">getSzekek_szama()</a>                             | 37 |
| 4.6.3.3.  | <a href="#">read()</a>  | 37 |
| 4.6.3.4.  | <a href="#">reserveHely()</a>                                 | 37 |
| 4.6.3.5.  | <a href="#">tele()</a>  | 38 |
| 4.6.3.6.  | <a href="#">write()</a>                                       | 38 |
| 4.6.4.    | <a href="#">Adattagok dokumentációja</a>                      | 38 |
| 4.6.4.1.  | <a href="#">foglalt_szekek_szama</a>                          | 38 |
| 4.6.4.2.  | <a href="#">kocsi_azonosito</a>                               | 38 |
| 4.6.4.3.  | <a href="#">szekek_szama</a>                                  | 38 |
| 4.7.      | <a href="#">Menetrend osztályreferencia</a>                   | 39 |
| 4.7.1.    | <a href="#">Részletes leírás</a>                              | 39 |
| 4.7.2.    | <a href="#">Konstruktorok és destruktorkok dokumentációja</a> | 40 |
| 4.7.2.1.  | <a href="#">Menetrend() [1/4]</a>                             | 40 |
| 4.7.2.2.  | <a href="#">Menetrend() [2/4]</a>                             | 40 |
| 4.7.2.3.  | <a href="#">Menetrend() [3/4]</a>                             | 40 |
| 4.7.2.4.  | <a href="#">Menetrend() [4/4]</a>                             | 40 |
| 4.7.2.5.  | <a href="#">~Menetrend()</a>                                  | 41 |
| 4.7.3.    | <a href="#">Tagfüggvények dokumentációja</a>                  | 41 |
| 4.7.3.1.  | <a href="#">addVonat()</a>                                    | 41 |
| 4.7.3.2.  | <a href="#">changeVonat()</a>                                 | 41 |
| 4.7.3.3.  | <a href="#">clear()</a>                                       | 41 |
| 4.7.3.4.  | <a href="#">createJegy()</a>                                  | 41 |
| 4.7.3.5.  | <a href="#">getVonat()</a>                                    | 42 |
| 4.7.3.6.  | <a href="#">getVonatokSzama()</a>                             | 42 |
| 4.7.3.7.  | <a href="#">kiir()</a>  | 42 |
| 4.7.3.8.  | <a href="#">read()</a>  | 43 |
| 4.7.3.9.  | <a href="#">removeVonat()</a>                                 | 43 |
| 4.7.3.10. | <a href="#">write()</a>                                       | 43 |
| 4.7.4.    | <a href="#">Adattagok dokumentációja</a>                      | 44 |
| 4.7.4.1.  | <a href="#">vonatok</a>                                       | 44 |
| 4.7.4.2.  | <a href="#">vonatok_szama</a>                                 | 44 |
| 4.8.      | <a href="#">Serializable osztályreferencia</a>                | 44 |
| 4.8.1.    | <a href="#">Részletes leírás</a>                              | 44 |
| 4.8.2.    | <a href="#">Tagfüggvények dokumentációja</a>                  | 44 |
| 4.8.2.1.  | <a href="#">read()</a>  | 44 |
| 4.8.2.2.  | <a href="#">write()</a>                                       | 45 |
| 4.9.      | <a href="#">Utvonal osztályreferencia</a>                     | 45 |
| 4.9.1.    | <a href="#">Részletes leírás</a>                              | 46 |
| 4.9.2.    | <a href="#">Konstruktorok és destruktorkok dokumentációja</a> | 46 |
| 4.9.2.1.  | <a href="#">Utvonal() [1/2]</a>                               | 46 |
| 4.9.2.2.  | <a href="#">Utvonal() [2/2]</a>                               | 46 |
| 4.9.2.3.  | <a href="#">~Utvonal()</a>                                    | 47 |

|            |  |    |
|------------|--|----|
| 4.9.3.     | Tagfüggvények dokumentációja .....                 | 47 |
| 4.9.3.1.   | addAllomas() .....                                 | 47 |
| 4.9.3.2.   | changeAllomas() .....                              | 47 |
| 4.9.3.3.   | createAllomas() .....                              | 47 |
| 4.9.3.4.   | getAllomas() .....                                 | 48 |
| 4.9.3.5.   | getAllomasokSzama() .....                          | 48 |
| 4.9.3.6.   | kiir() .....                                       | 48 |
| 4.9.3.7.   | operator=() .....                                  | 48 |
| 4.9.3.8.   | read() .....                                       | 48 |
| 4.9.3.9.   | removeAllomas() .....                              | 49 |
| 4.9.3.10.  | write() .....                                      | 49 |
| 4.9.4.     | Adattagok dokumentációja .....                     | 49 |
| 4.9.4.1.   | allomasok .....                                    | 49 |
| 4.9.4.2.   | allomasok_szama .....                              | 49 |
| 4.9.4.3.   | utvonal_azonosito .....                            | 50 |
| 4.10.      | Vonat osztályreferencia .....                      | 50 |
| 4.10.1.    | Részletes leírás .....                             | 51 |
| 4.10.2.    | Konstruktorok és destruktorok dokumentációja ..... | 51 |
| 4.10.2.1.  | Vonat() [1/3] .....                                | 51 |
| 4.10.2.2.  | Vonat() [2/3] .....                                | 51 |
| 4.10.2.3.  | Vonat() [3/3] .....                                | 51 |
| 4.10.2.4.  | ~Vonat() .....                                     | 52 |
| 4.10.3.    | Tagfüggvények dokumentációja .....                 | 52 |
| 4.10.3.1.  | addJegy() .....                                    | 52 |
| 4.10.3.2.  | addKocsi() .....                                   | 52 |
| 4.10.3.3.  | createJegy() .....                                 | 53 |
| 4.10.3.4.  | createKocsi() .....                                | 53 |
| 4.10.3.5.  | findAllomas() .....                                | 53 |
| 4.10.3.6.  | getAzonosito() .....                               | 54 |
| 4.10.3.7.  | getJegy() .....                                    | 54 |
| 4.10.3.8.  | getUtvonal() .....                                 | 54 |
| 4.10.3.9.  | indulasidoKulonbseg() .....                        | 54 |
| 4.10.3.10. | operator=() .....                                  | 55 |
| 4.10.3.11. | read() .....                                       | 55 |
| 4.10.3.12. | routeExists() .....                                | 55 |
| 4.10.3.13. | setAzonosito() .....                               | 56 |
| 4.10.3.14. | setUtvonal() .....                                 | 56 |
| 4.10.3.15. | write() .....                                      | 56 |
| 4.10.4.    | Adattagok dokumentációja .....                     | 57 |
| 4.10.4.1.  | jegyek .....                                       | 57 |
| 4.10.4.2.  | jegyek_szama .....                                 | 57 |
| 4.10.4.3.  | kocsik .....                                       | 57 |

|   |           |
|---|-----------|
| 4.10.4.4. kocsik_szama .....              | 57        |
| 4.10.4.5. utvonal .....                   | 57        |
| 4.10.4.6. vonat_azonosito .....           | 57        |
| <b>5. Fájlok dokumentációja</b> .....     | <b>59</b> |
| 5.1. allomas.cpp fájlreferencia .....     | 59        |
| 5.2. allomas.h fájlreferencia .....       | 59        |
| 5.3. allomas.h .....                      | 59        |
| 5.4. ido.cpp fájlreferencia .....         | 60        |
| 5.5. ido.h fájlreferencia .....           | 60        |
| 5.6. ido.h .....                          | 61        |
| 5.7. jegy.cpp fájlreferencia .....        | 61        |
| 5.8. jegy.h fájlreferencia .....          | 61        |
| 5.9. jegy.h .....                         | 62        |
| 5.10. kocsi.cpp fájlreferencia .....      | 63        |
| 5.11. kocsi.h fájlreferencia .....        | 63        |
| 5.12. kocsi.h .....                       | 64        |
| 5.13. main.cpp fájlreferencia .....       | 64        |
| 5.13.1. Függvények dokumentációja .....   | 64        |
| 5.13.1.1. main() .....                    | 64        |
| 5.14. menetrend.cpp fájlreferencia .....  | 65        |
| 5.15. menetrend.h fájlreferencia .....    | 65        |
| 5.16. menetrend.h .....                   | 65        |
| 5.17. serializable.h fájlreferencia ..... | 66        |
| 5.18. serializable.h .....                | 66        |
| 5.19. test.cpp fájlreferencia .....       | 66        |
| 5.19.1. Függvények dokumentációja .....   | 66        |
| 5.19.1.1. test() .....                    | 66        |
| 5.20. test.h fájlreferencia .....         | 73        |
| 5.20.1. Függvények dokumentációja .....   | 73        |
| 5.20.1.1. test() .....                    | 73        |
| 5.21. test.h .....                        | 80        |
| 5.22. utvonal.cpp fájlreferencia .....    | 80        |
| 5.23. utvonal.h fájlreferencia .....      | 80        |
| 5.24. utvonal.h .....                     | 80        |
| 5.25. vonat.cpp fájlreferencia .....      | 81        |
| 5.26. vonat.h fájlreferencia .....        | 81        |
| 5.27. vonat.h .....                       | 81        |
| <b>Tárgymutató</b> .....                  | <b>83</b> |

# 1. fejezet

## Hierarchikus mutató

### 1.1. Osztályhierarchia

Majdnem (de nem teljesen) betűrendbe szedett leszármazási lista:

|                            |    |
|----------------------------|----|
| Serializable . . . . .     | 44 |
| Allomas . . . . .          | 7  |
| Ido . . . . .              | 18 |
| Jegy . . . . .             | 24 |
| FelarasJegy . . . . .      | 14 |
| KedvezmenyesJegy . . . . . | 31 |
| Kocsi . . . . .            | 35 |
| Menetrend . . . . .        | 39 |
| Utvonal . . . . .          | 45 |
| Vonat . . . . .            | 50 |



## 2. fejezet

# Osztálymutató

### 2.1. Osztálylista

Az összes osztály, struktúra, unió és interfész listája rövid leírásokkal:

|  |    |
|--|----|
| Allomas  |    |
| Időtaroláshoz .....  | 7  |
| FelarasJegy .....  | 14 |
| Ido  |    |
| Szabványos bemenet és kimenet kezeléséhez .....                    | 18 |
| Jegy   |    |
| Szabványos bemenet és kimenet kezeléséhez .....                    | 24 |
| KedvezményesJegy .....   | 31 |
| Kocsi  |    |
| Szabványos bemenet és kimenet kezeléséhez .....                    | 35 |
| Menetrend  |    |
| Szabványos bemenet és kimenet kezeléséhez .....                    | 39 |
| Serializable   |    |
| Ostream és istream használatához .....                             | 44 |
| Utvonal  |    |
| Interfész az objektumok sorosításához és deszerializálásához ..... | 45 |
| Vonat .....  | 50 |





## 3. fejezet

# Fájlmutató

### 3.1. Fájllista

Az összes fájl listája rövid leírásokkal:

|                      |    |
|----------------------|----|
| allomas.cpp .....    | 59 |
| allomas.h .....      | 59 |
| ido.cpp .....        | 60 |
| ido.h .....          | 60 |
| jegy.cpp .....       | 61 |
| jegy.h .....         | 61 |
| kocsi.cpp .....      | 63 |
| kocsi.h .....        | 63 |
| main.cpp .....       | 64 |
| menetrend.cpp .....  | 65 |
| menetrend.h .....    | 65 |
| serializable.h ..... | 66 |
| test.cpp .....       | 66 |
| test.h .....         | 73 |
| utvonal.cpp .....    | 80 |
| utvonal.h .....      | 80 |
| vonat.cpp .....      | 81 |
| vonat.h .....        | 81 |



## 4. fejezet

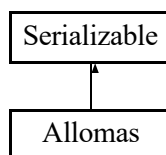
# Osztályok dokumentációja

### 4.1. Allomas osztályreferencia

Időtárolásához.

```
#include <allomas.h>
```

Az Allomas osztály származási diagramja:



#### Publikus tagfüggvények

- [Allomas](#) ()  
*Az érkezési időpont tárolója.*
- [Allomas](#) (const [Allomas](#) &other)
- [Allomas](#) & [operator=](#) (const [Allomas](#) &other)  
*Értékkadó operátor.*
- [Allomas](#) (int azonosito, const std::string &[nev](#), int erkezes\_ora, int erkezes\_perc, int indulas\_ora, int indulas\_←\_perc)
- void [changeErkezes](#) (int ora, int perc)
- void [changeIndulas](#) (int ora, int perc)
- void [changeNev](#) (const std::string &other)
- int [getIndulasOra](#) ()
- int [getIndulasPerc](#) ()
- int [getErkezesOra](#) ()
- int [getErkezesPerc](#) ()
- [Ido](#) & [getIndulas](#) ()
- [Ido](#) & [getErkezes](#) ()
- std::string & [getNev](#) ()
- void [write](#) (std::ostream &os) const
- void [read](#) (std::istream &is)
- [~Allomas](#) ()  
*Az osztály destruktora.*

## Privát attribútumok

- `int allomas_azonosito`
- `std::string nev`  
*Az állomás azonosítója.*
- `Ido indulas`  
*Az állomás neve.*
- `Ido erkezes`  
*Az indulási időpont tárolója.*

### 4.1.1. Részletes leírás

Időtárolásához.

Szabványos bemenet és kimenet kezeléséhez. Memória helyes használatának ellenőrzéséhez. Interfész az objektumok sorosításához és deszerializálásához. Az `Allomas` osztály definíciója.

### 4.1.2. Konstruktorkok és destruktorkok dokumentációja

#### 4.1.2.1. `Allomas()` [1/3]

```
Allomas::Allomas ( )
```

Az érkezési időpont tárolója.

Az osztály alapértelmezett konstruktora.

#### 4.1.2.2. `Allomas()` [2/3]

```
Allomas::Allomas (
    const Allomas & other )
```

Másoló konstruktor.

#### Paraméterek

|              |   |
|--------------|---|
| <i>other</i> | - Másolni kívánt <code>Allomas</code> objektum. |
|--------------|---|

#### 4.1.2.3. `Allomas()` [3/3]

```
Allomas::Allomas (
    int azonosito,
    const std::string & nev,
    int erkezes_ora,
    int erkezes_perc,
    int indulas_ora,
    int indulas_perc )
```

Egyedi konstruktor.

**Paraméterek**

|                     |                           |
|---------------------|---------------------------|
| <i>azonosito</i>    | - Az állomás azonosítója. |
| <i>nev</i>          | - Az állomás neve.        |
| <i>erkezes_ora</i>  | - Az érkezés órája.       |
| <i>erkezes_perc</i> | - Az érkezés perce.       |
| <i>indulas_ora</i>  | - Az indulás órája.       |
| <i>indulas_perc</i> | - Az indulás perce.       |

**4.1.2.4. ~Allomas()**

```
Allomas::~Allomas ( )
```

Az osztály destruktora.

**4.1.3. Tagfüggvények dokumentációja****4.1.3.1. changeErkezes()**

```
void Allomas::changeErkezes (
    int ora,
    int perc )
```

Az érkezés idejének módosítása.

**Paraméterek**

|             |                        |
|-------------|------------------------|
| <i>ora</i>  | - Az új érkezés órája. |
| <i>perc</i> | - Az új érkezés perce. |

**4.1.3.2. changeIndulas()**

```
void Allomas::changeIndulas (
    int ora,
    int perc )
```

Az indulás idejének módosítása.

**Paraméterek**

|             |                        |
|-------------|------------------------|
| <i>ora</i>  | - Az új indulás órája. |
| <i>perc</i> | - Az új indulás perce. |

#### 4.1.3.3. changeNev()

```
void Allomas::changeNev (
    const std::string & other )
```

Az állomás nevének módosítása.

##### Paraméterek

|              |              |
|--------------|--------------|
| <i>other</i> | - Az új név. |
|--------------|--------------|

#### 4.1.3.4. getErkezes()

```
Ido & Allomas::getErkezes ( )
```

Az érkezési időreferenciájának lekérdezése.

##### Visszatérési érték

Az érkezés idejének referenciája.

#### 4.1.3.5. getErkezesOra()

```
int Allomas::getErkezesOra ( )
```

Az érkezés órájának lekérdezése.

##### Visszatérési érték

Az érkezés órája.

#### 4.1.3.6. getErkezesPerc()

```
int Allomas::getErkezesPerc ( )
```

Az érkezés percének lekérdezése.

##### Visszatérési érték

Az érkezés perce.

#### 4.1.3.7. getIndulas()

```
Ido & Allomas::getIndulas ( )
```

Az indulási időreferenciájának lekérdezése.

##### Visszatérési érték

Az indulás idejének referenciája.

#### 4.1.3.8. getIndulasOra()

```
int Allomas::getIndulasOra ( )
```

Az indulás órájának lekérdezése.

##### Visszatérési érték

Az indulás órája.

#### 4.1.3.9. getIndulasPerc()

```
int Allomas::getIndulasPerc ( )
```

Az indulás percének lekérdezése.

##### Visszatérési érték

Az indulás perce.

#### 4.1.3.10. getNev()

```
std::string & Allomas::getNev ( )
```

Az állomás névének lekérdezése.

##### Visszatérési érték

Az állomás neve.

#### 4.1.3.11. operator=()

```
Allomas & Allomas::operator= (
    const Allomas & other )
```

Értékadó operátor.

#### 4.1.3.12. read()

```
void Allomas::read (
    std::istream & is ) [virtual]
```

[Serializable](#) interfész implementációja: objektum olvasása adatfolyamból.

##### Paraméterek

|           |   |
|-----------|---|
| <i>is</i> | - A beolvasásra használt bemeneti adatfolyam. |
|-----------|---|



Megvalósítja a következőket: [Serializable](#).

#### 4.1.3.13. write()

```
void Allomas::write (
    std::ostream & os ) const [virtual]
```

[Serializable](#) interfész implementációja: objektum írása adatfolyamba.

##### Paraméterek

|    |   |
|----|---|
| os | - Az írásra használt kimeneti adatfolyam. |
|----|---|

Megvalósítja a következőket: [Serializable](#).

#### 4.1.4. Adattagok dokumentációja

##### 4.1.4.1. allomas\_azonosito

```
int Allomas::allomas_azonosito [private]
```

##### 4.1.4.2. erkezes

```
Ido Allomas::erkezes [private]
```

Az indulási időpont tárolója.

##### 4.1.4.3. indulas

```
Ido Allomas::indulas [private]
```

Az állomás neve.

##### 4.1.4.4. nev

```
std::string Allomas::nev [private]
```

Az állomás azonosítója.

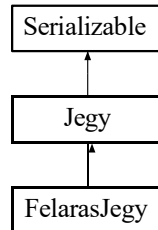
Ez a dokumentáció az osztályról a következőfájlok alapján készült:

- [allomas.h](#)
- [allomas.cpp](#)

## 4.2. FelarasJegy osztályreferencia

```
#include <jegy.h>
```

A FelarasJegy osztály származási diagramja:



### Publikus tagfüggvények

- `FelarasJegy ()`  
*A felár értékét.*
- `FelarasJegy (const FelarasJegy &other)`
- `FelarasJegy (int azonosito, int hely, int kocsi, int vonat, const std::string &indulo, ldo indulo_idopont, const std::string &cel, ldo cel_idopont, double felar, std::string tipus)`
- `FelarasJegy & operator= (const FelarasJegy &other)`
- `void kiir (std::ostream &os) const`
- `void write (std::ostream &os) const`
- `void read (std::istream &is)`
- `Jegy * clone () const`
- `~FelarasJegy ()`  
*Az osztály destruktora.*

### Publikus tagfüggvények a(z) `Jegy` osztályból származnak

- `Jegy ()`  
*A jegy árát számító függvény.*
- `Jegy (int azonosito, int hely, int kocsi, int vonat, const std::string &indulo, ldo indulo_idopont, const std::string &cel, ldo cel_idopont)`
- `Jegy (int azonosito, int hely, int kocsi, int vonat, const std::string &indulo, ldo indulo_idopont, const std::string &cel, ldo cel_idopont, double pred)`
- `Jegy (const Jegy &other)`
- `Jegy & operator= (const Jegy &other)`
- `bool operator== (const Jegy &other) const`
- `bool operator!= (const Jegy &other) const`
- `void kiir (std::ostream &os) const`
- `void write (std::ostream &os) const`
- `void read (std::istream &is)`
- `virtual ~Jegy ()`  
*Az osztály virtuális destruktora.*

### Privát attribútumok

- `std::string tipus`
- `double felar`  
*A féláras típus neve.*

### További örökölt tagok

### Védett tagfüggvények a(z) **Jegy** osztályból származnak

- virtual int **getAr** (double pred)  
*A jegy ára.*

### Védett attribútumok a(z) **Jegy** osztályból származnak

- int **jegy\_azonosito**
- int **helyszam**  
*A jegy azonosítója.*
- int **kocsi\_szam**  
*A jegyhez tartozó helyszám.*
- int **vonat\_szam**  
*A kocsi azonosítója, ahol a hely található.*
- std::string **indulo\_nev**  
*A vonat azonosítója, amelyre a jegy szól.*
- ldo **indulo\_ido**  
*Az indulási állomás neve.*
- std::string **cel\_nev**  
*Az indulási időpont.*
- ldo **cel\_ido**  
*A célállomás neve.*
- int **ar**  
*A célállomás időpontja.*

#### 4.2.1. Részletes leírás

A FélarasJegy osztály a **Jegy** osztály leszármazottja, és reprezentálja a félaras jegyeket. Tartalmazza a típust és a felárat.

#### 4.2.2. Konstruktorkok és destruktorkok dokumentációja

##### 4.2.2.1. FelarasJegy() [1/3]

```
FelarasJegy::FelarasJegy ( )
```

A felár értéket.

Az osztály alapértelmezett konstruktora.

##### 4.2.2.2. FelarasJegy() [2/3]

```
FelarasJegy::FelarasJegy (
    const FelarasJegy & other )
```

Másoló konstruktor.

**Paraméterek**

|              |                                       |
|--------------|---------------------------------------|
| <i>other</i> | - másolni kívánt FélarasJegy objektum |
|--------------|---------------------------------------|

**4.2.2.3. FelarasJegy() [3/3]**

```

FelarasJegy::FelarasJegy (
    int azonosito,
    int hely,
    int kocsi,
    int vonat,
    const std::string & indulo,
    Idő indulo_idopont,
    const std::string & cel,
    Idő cel_idopont,
    double felar = 0.6,
    std::string tipus = "felaras jegy" )

```

Paraméteres konstruktor.

**Paraméterek**

|                  |                        |
|------------------|------------------------|
| <i>azonosito</i> | - a jegy azonosítója   |
| <i>hely</i>      | - a helyszám           |
| <i>kocsi</i>     | - a kocsi azonosítója  |
| <i>vonat</i>     | - a vonat azonosítója  |
| <i>indulo</i>    | - az indulási állomás  |
| <i>cel</i>       | - a célállomás         |
| <i>felar</i>     | - a felár értéket      |
| <i>tipus_nev</i> | - a félaras típus neve |

**4.2.2.4. ~FelarasJegy()**

```

FelarasJegy::~~FelarasJegy ( )

```

Az osztály destruktora.

**4.2.3. Tagfüggvények dokumentációja****4.2.3.1. clone()**

```

Jegy & FelarasJegy::clone ( ) const [virtual]

```

Létrehoz egy új **Jegy** objektum másolatot az aktuális objektumról.

**Visszatérési érték**

**Jegy** objektumra mutató pointer, amely az új másolatot tartalmazza.

Újrimplementált ősök: **Jegy**.

#### 4.2.3.2. kiir()

```
void FelarasJegy::kiir (
    std::ostream & os ) const
```

Objektum kiírása adatfolyamba esztétikus formában.

##### Paraméterek

|    |   |
|----|---|
| os | - Az írásra használt kimeneti adatfolyam. |
|----|---|

#### 4.2.3.3. operator=()

```
FelarasJegy & FelarasJegy::operator= (
    const FelarasJegy & other )
```

Értékadó operátor.

##### Paraméterek

|       |                                       |
|-------|---------------------------------------|
| other | - másolni kívánt FélarasJegy objektum |
|-------|---------------------------------------|

##### Visszatérési érték

A másolt FélarasJegy objektum referenciája.

#### 4.2.3.4. read()

```
void FelarasJegy::read (
    std::istream & is ) [virtual]
```

[Serializable](#) interfész implementációja: objektum olvasása adatfolyamból.

##### Paraméterek

|    |   |
|----|---|
| is | - A beolvasásra használt bemeneti adatfolyam. |
|----|---|

Megvalósítja a következőket: [Serializable](#).

#### 4.2.3.5. write()

```
void FelarasJegy::write (
    std::ostream & os ) const [virtual]
```

[Serializable](#) interfész implementációja: objektum írása adatfolyamba.

### Paraméterek

|    |   |
|----|---|
| os | - Az írásra használt kimeneti adatfolyam. |
|----|---|

Megvalósítja a következőket: [Serializable](#).

## 4.2.4. Adattagok dokumentációja

### 4.2.4.1. felar

```
double FelarasJegy::felar [private]
```

A féláras típus neve.

### 4.2.4.2. tipus

```
std::string FelarasJegy::tipus [private]
```

Ez a dokumentáció az osztályról a következőfájlok alapján készült:

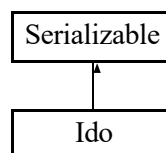
- [jegy.h](#)
- [jegy.cpp](#)

## 4.3. Ido osztályreferencia

Szabványos bemenet és kimenet kezeléséhez.

```
#include <ido.h>
```

Az Ido osztály származási diagramja:



### Publikus tagfüggvények

- [Ido](#) ()  
*Az időobjektumban tárolt perc. (0-59)*
- [Ido](#) (int o, int p)
- [Ido](#) (const [Ido](#) &other)
- int [getOra](#) () const
- int [getPerc](#) () const
- void [setOra](#) (int o)
- void [setPerc](#) (int p)
- void [setIdo](#) (int o, int p)
- void [addPerc](#) (int p)
- void [addOra](#) (int o)
- bool [operator==](#) (const [Ido](#) &other) const
- bool [operator!=](#) (const [Ido](#) &other) const
- void [kiir](#) (std::ostream &os) const
- void [write](#) (std::ostream &os) const
- void [read](#) (std::istream &is)
- [~Ido](#) ()  
*Az osztály destruktora.*

**Privát attribútumok**

- int `ora`
- int `perc`

*Az időobjektumban tárolt óra. (0-23)*

**4.3.1. Részletes leírás**

Szabványos bemenet és kimenet kezeléséhez.

Interfész az objektumok sorosításához és deszerializálásához. Memória helyes használatának ellenőrzéséhez. Az Időosztály reprezentál egy időpontot. Tartalmazza az időpontot alkotó órát és percet.

**4.3.2. Konstruktorok és destruktorok dokumentációja****4.3.2.1. Ido()** [1/3]

```
Ido::Ido ( )
```

Az időobjektumban tárolt perc. (0-59) Az

osztály alapértelmezett konstruktora.

**4.3.2.2. Ido()** [2/3]

```
Ido::Ido (
    int o,
    int p )
```

Paraméteres konstruktor.

**Paraméterek**

|          |                          |
|----------|--------------------------|
| <i>o</i> | - A tárolni kívánt óra.  |
| <i>p</i> | - A tárolni kívánt perc. |

**4.3.2.3. Ido()** [3/3]

```
Ido::Ido (
    const Ido & other )
```

Az osztály másoló konstruktora.

**Paraméterek**

|              |   |
|--------------|---|
| <i>other</i> | - Másolni kívánt <code>Ido</code> objektum. |
|--------------|---|

#### 4.3.2.4. ~Ido()

```
Ido::~~Ido ( )
```

Az osztály destruktora.

### 4.3.3. Tagfüggvények dokumentációja

#### 4.3.3.1. addOra()

```
void Ido::addOra (
    int o )
```

Óra hozzáadása az aktuális időhöz.

##### Paraméterek

|          |                             |
|----------|-----------------------------|
| <i>o</i> | - A hozzáadandó óra értéke. |
|----------|-----------------------------|

#### 4.3.3.2. addPerc()

```
void Ido::addPerc (
    int p )
```

Perc hozzáadása az aktuális időhöz.

##### Paraméterek

|          |                              |
|----------|------------------------------|
| <i>p</i> | - A hozzáadandó perc értéke. |
|----------|------------------------------|

#### 4.3.3.3. getOra()

```
int Ido::getOra ( ) const
```

Getter függvény az óra lekérdezéséhez.

##### Visszatérési érték

Az aktuális óra értéke.

#### 4.3.3.4. getPerc()

```
int Ido::getPerc ( ) const
```

Getter függvény a perc lekérdezéséhez.

##### Visszatérési érték

Az aktuális perc értéke.



#### 4.3.3.5. kiir()

```
void Ido::kiir (
    std::ostream & os ) const
```

Objektum kiírása adatfolyamba esztétikus formában.

##### Paraméterek

|           |   |
|-----------|---|
| <i>os</i> | - Az írásra használt kimeneti adatfolyam. |
|-----------|---|

#### 4.3.3.6. operator"!="()

```
bool Ido::operator!= (
    const Ido & other ) const
```

Nem egyenlőség operátor.

##### Paraméterek

|              |  |
|--------------|--|
| <i>other</i> | - A másik idő, amivel a nem egyenlőség vizsgálat történik. |
|--------------|--|

##### Visszatérési érték

igaz, ha az idők nem egyeznek meg, különben hamis.

#### 4.3.3.7. operator==( )

```
bool Ido::operator== (
    const Ido & other ) const
```

Egyenlőség operátor.

##### Paraméterek

|              |   |
|--------------|---|
| <i>other</i> | - A másik idő, amivel az egyenlőség vizsgálat történik. |
|--------------|---|

##### Visszatérési érték

igaz, ha az idők megegyeznek, különben hamis.

#### 4.3.3.8. read()

```
void Ido::read (
    std::istream & is ) [virtual]
```

[Serializable](#) interfész implementációja: objektum olvasása adatfolyamból.

**Paraméterek**

|           |   |
|-----------|---|
| <i>is</i> | - A beolvasásra használt bemeneti adatfolyam. |
|-----------|---|

Megvalósítja a következőket: [Serializable](#).

**4.3.3.9. setIdo()**

```
void Ido::setIdo (
    int o,
    int p )
```

Időbeállítása adott óra és perc értékekkel.

**Paraméterek**

|          |                              |
|----------|------------------------------|
| <i>o</i> | - A beállítandó óra értéke.  |
| <i>p</i> | - A beállítandó perc értéke. |

**4.3.3.10. setOra()**

```
void Ido::setOra (
    int o )
```

Setter függvény az óra beállításához.

**Paraméterek**

|          |                             |
|----------|-----------------------------|
| <i>o</i> | - A beállítandó óra értéke. |
|----------|-----------------------------|

**4.3.3.11. setPerc()**

```
void Ido::setPerc (
    int p )
```

Setter függvény a perc beállításához.

**Paraméterek**

|          |                              |
|----------|------------------------------|
| <i>p</i> | - A beállítandó perc értéke. |
|----------|------------------------------|

**4.3.3.12. write()**

```
void Ido::write (
    std::ostream & os ) const [virtual]
```

[Serializable](#) interfész implementációja: objektum írása adatfolyamba.

### Paraméterek

|    |   |
|----|---|
| os | - Az írásra használt kimeneti adatfolyam. |
|----|---|

Megvalósítja a következőket: [Serializable](#).

## 4.3.4. Adattagok dokumentációja

### 4.3.4.1. ora

```
int Idó::ora [private]
```

### 4.3.4.2. perc

```
int Idó::perc [private]
```

Az időobjektumban tárolt óra. (0-23)

Ez a dokumentáció az osztályról a következőfájlok alapján készült:

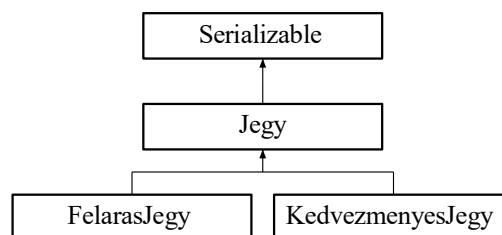
- [ido.h](#)
- [ido.cpp](#)

## 4.4. Jegy osztályreferencia

Szabványos bemenet és kimenet kezeléséhez.

```
#include <jegy.h>
```

A Jegy osztály származási diagramja:



### Publikus tagfüggvények

- `Jegy ()`  
*A jegy árát számító függvény.*
- `Jegy (int azonosito, int hely, int kocsi, int vonat, const std::string &indulo, ldo indulo_idopont, const std::string &cel, ldo cel_idopont)`
- `Jegy (int azonosito, int hely, int kocsi, int vonat, const std::string &indulo, ldo indulo_idopont, const std::string &cel, ldo cel_idopont, double pred)`
- `Jegy (const Jegy &other)`
- `Jegy & operator= (const Jegy &other)`
- `bool operator== (const Jegy &other) const`
- `bool operator!= (const Jegy &other) const`
- `void kiir (std::ostream &os) const`
- `void write (std::ostream &os) const`
- `void read (std::istream &is)`
- `virtual Jegy * clone () const`
- `virtual ~Jegy ()`  
*Az osztály virtuális destruktora.*

### Védett tagfüggvények

- `virtual int getAr (double pred)`  
*A jegy ára.*

### Védett attribútumok

- `int jegy_azonosito`
- `int helyszam`  
*A jegy azonosítója.*
- `int kocsi_szam`  
*A jegyhez tartozó helyszám.*
- `int vonat_szam`  
*A kocsi azonosítója, ahol a hely található.*
- `std::string indulo_nev`  
*A vonat azonosítója, amelyre a jegy szól.*
- `ldo indulo_ido`  
*Az indulási állomás neve.*
- `std::string cel_nev`  
*Az indulási időpont.*
- `ldo cel_ido`  
*A célállomás neve.*
- `int ar`  
*A célállomás időpontja.*

#### 4.4.1. Részletes leírás

Szabványos bemenet és kimenet kezeléséhez.

Interfész az objektumok sorosításához és deszerializálásához. Memória helyes használatának ellenőrzéséhez. Időtárolásához. A `Jegy` osztály reprezentálja a vonatjegyeket. Tartalmazza a jegy azonosítóját, a helyszámot, a kocsi számát, a vonat számát, az indulo es a cel allomast, valamint az arat.

## 4.4.2. Konstruktorok és destruktorok dokumentációja

### 4.4.2.1. Jegy() [1/4]

```
Jegy::Jegy ( )
```

A jegy árát számító függvény.

Az osztály alapertelmezett konstruktora.

### 4.4.2.2. Jegy() [2/4]

```
Jegy::Jegy (
    int azonosito,
    int hely,
    int kocsi,
    int vonat,
    const std::string & indulo,
    Idó indulo_idopont,
    const std::string & cel,
    Idó cel_idopont )
```

Paraméteres konstruktor.

#### Paraméterek

|                  |                       |
|------------------|-----------------------|
| <i>azonosito</i> | - a jegy azonosítója  |
| <i>hely</i>      | - a helyszám          |
| <i>kocsi</i>     | - a kocsi azonosítója |
| <i>vonat</i>     | - a vonat azonosítója |
| <i>indulo</i>    | - az indulási állomás |
| <i>cel</i>       | - a célállomás        |

### 4.4.2.3. Jegy() [3/4]

```
Jegy::Jegy (
    int azonosito,
    int hely,
    int kocsi,
    int vonat,
    const std::string & indulo,
    Idó indulo_idopont,
    const std::string & cel,
    Idó cel_idopont,
    double pred )
```

Paraméteres konstruktor (alosztályhoz).

#### Paraméterek

|                  |                      |
|------------------|----------------------|
| <i>azonosito</i> | - a jegy azonosítója |
|------------------|----------------------|

**Paraméterek**

|               |  |
|---------------|--|
| <i>hely</i>   | - a helyszám                                       |
| <i>kocsi</i>  | - a kocsi azonosítója                              |
| <i>vonat</i>  | - a vonat azonosítója                              |
| <i>indulo</i> | - az indulási állomás                              |
| <i>cel</i>    | - a célállomás                                     |
| <i>pred</i>   | - az árat meghatározó kedvezmény vagy felár értéke |

**4.4.2.4. Jegy()** [4/4]

```
Jegy::Jegy (
    const Jegy & other )
```

Az osztály másoló konstruktora.

**Paraméterek**

|              |   |
|--------------|---|
| <i>other</i> | - Másolni kívánt <a href="#">Jegy</a> objektum. |
|--------------|---|

**4.4.2.5. ~Jegy()**

```
virtual Jegy::~Jegy ( ) [inline], [virtual]
```

Az osztály virtuális destruktora.

**4.4.3. Tagfüggvények dokumentációja****4.4.3.1. clone()**

```
Jegy & Jegy::clone ( ) const [virtual]
```

Létrehoz egy új [Jegy](#) objektum másolatot az aktuális objektumról.

**Visszatérési érték**

[Jegy](#) objektumra mutató pointer, amely az új másolatot tartalmazza.

Újrimplementáló leszármazottak: [KedvezmenyesJegy](#) és [FelarasJegy](#).

**4.4.3.2. getAr()**

```
int Jegy::getAr (
    double pred = 1 ) [protected], [virtual]
```

A jegy ára.

**4.4.3.3. kiir()**

```
void Jegy::kiir (
    std::ostream & os ) const
```

Objektum kiírása adatfolyamba esztétikus formában.

**Paraméterek**

|           |   |
|-----------|---|
| <i>os</i> | - Az írásra használt kimeneti adatfolyam. |
|-----------|---|

**4.4.3.4. operator"!=()**

```
bool Jegy::operator!= (
    const Jegy & other ) const
```

Nem egyenlőség operátor.

**Paraméterek**

|              |   |
|--------------|---|
| <i>other</i> | - A másik jegy, amivel a nem egyenlőség vizsgálat történik. |
|--------------|---|

**Visszatérési érték**

igaz, ha a jegyek nem egyeznek meg, különben hamis.

**4.4.3.5. operator=()**

```
Jegy & Jegy::operator= (
    const Jegy & other )
```

Értékadó operátor.

**Paraméterek**

|              |  |
|--------------|--|
| <i>other</i> | - Másolni kívánt <i>Jegy</i> objektum. |
|--------------|--|

**Visszatérési érték**

A másolt *Jegy* objektum referenciája.

**4.4.3.6. operator==(())**

```
bool Jegy::operator== (
    const Jegy & other ) const
```

Egyenlőség operátor.

**Paraméterek**

|              |  |
|--------------|--|
| <i>other</i> | - A másik jegy, amivel az egyenlőség vizsgálat történik. |
|--------------|--|



**Visszatérési érték**

igaz, ha a jegyek megegyeznek, különben hamis.

**4.4.3.7. read()**

```
void Jegy::read (
    std::istream & is ) [virtual]
```

[Serializable](#) interfész implementációja: objektum olvasása adatfolyamból.

**Paraméterek**

|                 |   |
|-----------------|---|
| <code>is</code> | - A beolvasásra használt bemeneti adatfolyam. |
|-----------------|---|

Megvalósítja a következőket: [Serializable](#).

Újrimplementáló leszármazottak: [KedvezményesJegy](#).

**4.4.3.8. write()**

```
void Jegy::write (
    std::ostream & os ) const [virtual]
```

[Serializable](#) interfész implementációja: objektum írása adatfolyamba.

**Paraméterek**

|                 |   |
|-----------------|---|
| <code>os</code> | - Az írásra használt kimeneti adatfolyam. |
|-----------------|---|

Megvalósítja a következőket: [Serializable](#).

Újrimplementáló leszármazottak: [KedvezményesJegy](#).

**4.4.4. Adattagok dokumentációja****4.4.4.1. ar**

```
int Jegy::ar [protected]
```

A célállomás időpontja.

**4.4.4.2. cel\_ido**

```
Ido Jegy::cel_ido [protected]
```

A célállomás neve.

#### 4.4.4.3. cel\_nev

```
std::string Jegy::cel_nev [protected]
```

Az indulási időpont.

#### 4.4.4.4. helyszam

```
int Jegy::helyszam [protected]
```

A jegy azonosítója.

#### 4.4.4.5. indulo\_ido

```
Ido Jegy::indulo_ido [protected]
```

Az indulási állomás neve.

#### 4.4.4.6. indulo\_nev

```
std::string Jegy::indulo_nev [protected]
```

A vonat azonosítója, amelyre a jegy szól.

#### 4.4.4.7. jegy\_azonosito

```
int Jegy::jegy_azonosito [protected]
```

#### 4.4.4.8. kocsi\_szam

```
int Jegy::kocsi_szam [protected]
```

A jegyhez tartozó helyszám.

#### 4.4.4.9. vonat\_szam

```
int Jegy::vonat_szam [protected]
```

A kocsi azonosítója, ahol a hely található.

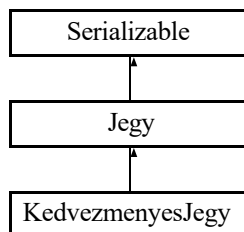
Ez a dokumentáció az osztályról a következőfájlok alapján készült:

- [jegy.h](#)
- [jegy.cpp](#)

## 4.5. KedvezmenyesJegy osztályreferencia

```
#include <jegy.h>
```

A KedvezmenyesJegy osztály származási diagramja:



### Publikus tagfüggvények

- `KedvezmenyesJegy ()`  
*A kedvezmény mértékét százalékban kifejezőérték.*
- `KedvezmenyesJegy (const KedvezmenyesJegy &other)`
- `KedvezmenyesJegy (int azonosito, int hely, int kocsi, int vonat, const std::string &indulo, ldo indulo_idopont, const std::string &cel, ldo cel_idopont, double kedvezmeny, std::string tipus)`
- `KedvezmenyesJegy & operator= (const KedvezmenyesJegy &other)`
- `void kiir (std::ostream &os) const`
- `void write (std::ostream &os) const`
- `void read (std::istream &is)`
- `Jegy * clone () const`
- `~KedvezmenyesJegy ()`  
*Az osztály destruktora.*

### Publikus tagfüggvények a(z) `Jegy` osztályból származnak

- `Jegy ()`  
*A jegy árát számító függvény.*
- `Jegy (int azonosito, int hely, int kocsi, int vonat, const std::string &indulo, ldo indulo_idopont, const std::string &cel, ldo cel_idopont)`
- `Jegy (int azonosito, int hely, int kocsi, int vonat, const std::string &indulo, ldo indulo_idopont, const std::string &cel, ldo cel_idopont, double pred)`
- `Jegy (const Jegy &other)`
- `Jegy & operator= (const Jegy &other)`
- `bool operator== (const Jegy &other) const`
- `bool operator!= (const Jegy &other) const`
- `void kiir (std::ostream &os) const`
- `virtual ~Jegy ()`  
*Az osztály virtuális destruktora.*

### Privát attribútumok

- `std::string tipus`
- `double kedvezmeny`  
*A kedvezményes típus neve.*

### További örökölt tagok

### Védett tagfüggvények a(z) **Jegy** osztályból származnak

- virtual int **getAr** (double pred)

*A jegy ára.*

### Védett attribútumok a(z) **Jegy** osztályból származnak

- int **jegy\_azonosito**

*A jegy azonosítója.*

- int **helyszam**

*A jegyhez tartozó helyszám.*

- int **kocsi\_szam**

*A kocsi azonosítója, ahol a hely található.*

- std::string **indulo\_nev**

*A vonat azonosítója, amelyre a jegy szól.*

- ldo **indulo\_ido**

*Az indulási állomás neve.*

- std::string **cel\_nev**

*Az indulási időpont.*

- ldo **cel\_ido**

*A célállomás neve.*

- int **ar**

*A célállomás időpontja.*

### 4.5.1. Részletes leírás

A KedvezményesJegy osztály a **Jegy** osztály leszármazottja, és reprezentálja a kedvezményes jegyeket. Tartalmazza a típust (pl. diák, nyugdíjas) és a kedvezmény mértékét.

### 4.5.2. Konstruktorkok és destruktorkok dokumentációja

#### 4.5.2.1. KedvezmenyesJegy() [1/3]

```
KedvezmenyesJegy::KedvezmenyesJegy ( )
```

A kedvezmény mértékét százalékban kifejezőérték.

Az osztály alapértelmezett konstruktora.

#### 4.5.2.2. KedvezmenyesJegy() [2/3]

```
KedvezmenyesJegy::KedvezmenyesJegy (
    const KedvezmenyesJegy & other )
```

Másoló konstruktor.

**Paraméterek**

|              |  |
|--------------|--|
| <i>other</i> | - másolni kívánt <a href="#">KedvezményesJegy</a> objektum |
|--------------|--|

**4.5.2.3. KedvezményesJegy() [3/3]**

```
KedvezményesJegy::KedvezményesJegy (
    int azonosito,
    int hely,
    int kocsi,
    int vonat,
    const std::string & indulo,
    Idő indulo_idopont,
    const std::string & cel,
    Idő cel_idopont,
    double kedvezmeny = -0.6,
    std::string tipus = "kedvezményes jegy" )
```

Paraméteres konstruktor.

**Paraméterek**

|                   |                                       |
|-------------------|---------------------------------------|
| <i>azonosito</i>  | - a jegy azonosítója                  |
| <i>hely</i>       | - a helyszám                          |
| <i>kocsi</i>      | - a kocsi azonosítója                 |
| <i>vonat</i>      | - a vonat azonosítója                 |
| <i>indulo</i>     | - az indulási állomás                 |
| <i>cel</i>        | - a célállomás                        |
| <i>kedvezmeny</i> | - a kedvezmény mértékét kifejezőérték |
| <i>tipus_nev</i>  | - a kedvezményes típus neve           |

**4.5.2.4. ~KedvezményesJegy()**

```
KedvezményesJegy::~~KedvezményesJegy ( )
```

Az osztály destruktora.

**4.5.3. Tagfüggvények dokumentációja****4.5.3.1. clone()**

```
Jegy & KedvezményesJegy::clone ( ) const [virtual]
```

Létrehoz egy új [Jegy](#) objektum másolatot az aktuális objektumról.

**Visszatérési érték**

[Jegy](#) objektumra mutató pointer, amely az új másolatot tartalmazza.

Újrimplementált ősök: [Jegy](#).

#### 4.5.3.2. kiir()

```
void KedvezmenyesJegy::kiir (
    std::ostream & os ) const
```

Objektum kiírása adatfolyamba esztétikus formában.

##### Paraméterek

|    |   |
|----|---|
| os | - Az írásra használt kimeneti adatfolyam. |
|----|---|

#### 4.5.3.3. operator=()

```
KedvezmenyesJegy & KedvezmenyesJegy::operator= (
    const KedvezmenyesJegy & other )
```

Értékadó operátor.

##### Paraméterek

|       |  |
|-------|--|
| other | - másolni kívánt <a href="#">KedvezmenyesJegy</a> objektum |
|-------|--|

##### Visszatérési érték

A másolt [KedvezmenyesJegy](#) objektum referenciája.

#### 4.5.3.4. read()

```
void KedvezmenyesJegy::read (
    std::istream & is ) [virtual]
```

[Serializable](#) interfész implementációja: objektum olvasása adatfolyamból.

##### Paraméterek

|    |   |
|----|---|
| is | - A beolvasásra használt bemeneti adatfolyam. |
|----|---|

Újraimplementált ősök: [Jegy](#).

#### 4.5.3.5. write()

```
void KedvezmenyesJegy::write (
    std::ostream & os ) const [virtual]
```

[Serializable](#) interfész implementációja: objektum írása adatfolyamba.

## Paraméterek

|    |   |
|----|---|
| os | - Az írásra használt kimeneti adatfolyam. |
|----|---|

Újraimplementált ősök: [Jegy](#).

#### 4.5.4. Adattagok dokumentációja

##### 4.5.4.1. kedvezmeny

```
double KedvezmenyesJegy::kedvezmeny [private]
```

A kedvezményes típus neve.

##### 4.5.4.2. tipus

```
std::string KedvezmenyesJegy::tipus [private]
```

Ez a dokumentáció az osztályról a következőfájlok alapján készült:

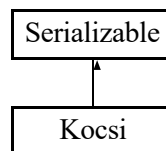
- [jegy.h](#)
- [jegy.cpp](#)

## 4.6. Kocsi osztályreferencia

Szabványos bemenet és kimenet kezeléséhez.

```
#include <kocsi.h>
```

A Kocsi osztály származási diagramja:



#### Publikus tagfüggvények

- [Kocsi](#) ()  
*A kocsihoz tartozó foglalt szekek szama.*
- [Kocsi](#) (const [Kocsi](#) &other)
- [Kocsi](#) (int azonosito, int szekek)
- bool [tele](#) () const
- int [reserveHely](#) ()
- int [getSzekek\\_szama](#) () const
- int [getAzonosito](#) () const
- void [write](#) (std::ostream &os) const
- void [read](#) (std::istream &is)
- [~Kocsi](#) ()  
*Az osztály destruktora.*

**Privát attribútumok**

- int [kocsi\\_azonosito](#)
- int [szekek\\_szama](#)  
*A kocsi azonosítója.*
- int [foglalt\\_szekek\\_szama](#)  
*A kocsihoz tartozó szekek száma.*

**4.6.1. Részletes leírás**

Szabványos bemenet és kimenet kezeléséhez.

Interfész az objektumok sorosításához és deszerializálásához. Memória helyes használatának ellenőrzéséhez. A [Kocsi](#) osztály reprezentálja a vonatkocsikat. Tartalmazza a kocsi azonosítóját, a szekek számát, valamint a foglalt szekek számát.

**4.6.2. Konstruktorkok és destruktorkok dokumentációja****4.6.2.1. Kocsi() [1/3]**

```
Kocsi::Kocsi ( )
```

A kocsihoz tartozó foglalt szekek száma.

Az osztály alapértelmezett konstruktora.

**4.6.2.2. Kocsi() [2/3]**

```
Kocsi::Kocsi (
    const Kocsi & other )
```

Az osztály másoló konstruktora.

**Paraméterek**

|              |  |
|--------------|--|
| <i>other</i> | - Másolni kívánt <a href="#">Kocsi</a> objektum. |
|--------------|--|

**4.6.2.3. Kocsi() [3/3]**

```
Kocsi::Kocsi (
    int azonosito,
    int szekek )
```

Egyedi konstruktor.

**Paraméterek**

|                  |   |
|------------------|---|
| <i>azonosito</i> | - A kocsi azonosítója.                    |
| <i>szekek</i>    | - A kocsiban található üres szekek száma. |



#### 4.6.2.4. ~Kocsi()

```
Kocsi::~~Kocsi ( )
```

Az osztály destruktora.

### 4.6.3. Tagfüggvények dokumentációja

#### 4.6.3.1. getAzonosito()

```
int Kocsi::getAzonosito ( ) const
```

Az osztály azonosítójának lekérdezése.

##### Visszatérési érték

A kocsi azonosítója.

#### 4.6.3.2. getSzekek\_szama()

```
int Kocsi::getSzekek_szama ( ) const
```

Székek számának lekérdezése.

##### Visszatérési érték

A kocsihoz tartozó székek száma.

#### 4.6.3.3. read()

```
void Kocsi::read (
    std::istream & is ) [virtual]
```

[Serializable](#) interfész implementációja: objektum olvasása adatfolyamból.

##### Paraméterek

|           |   |
|-----------|---|
| <i>is</i> | - A beolvasásra használt bemeneti adatfolyam. |
|-----------|---|

Megvalósítja a következőket: [Serializable](#).

#### 4.6.3.4. reserveHely()

```
int Kocsi::reserveHely ( )
```

Foglal egy helyet a kocsiiban, és visszaadja a hely azonosítóját.

**Visszatérési érték**

A foglalt hely azonosítója.

**4.6.3.5. tele()**

```
bool Kocsi::tele ( ) const
```

Megvizsgálja, hogy a kocsiban van-e üres szék.

**Visszatérési érték**

igaz, ha a kocsi tele van, különben hamis.

**4.6.3.6. write()**

```
void Kocsi::write (
    std::ostream & os ) const [virtual]
```

[Serializable](#) interfész implementációja: objektum írása adatfolyamba.

**Paraméterek**

|    |   |
|----|---|
| os | - Az írásra használt kimeneti adatfolyam. |
|----|---|

Megvalósítja a következőket: [Serializable](#).

**4.6.4. Adattagok dokumentációja****4.6.4.1. foglalt\_szekek\_szama**

```
int Kocsi::foglalt_szekek_szama [private]
```

A kocsihoz tartozó szekek szama.

**4.6.4.2. kocsi\_azonosito**

```
int Kocsi::kocsi_azonosito [private]
```

**4.6.4.3. szekek\_szama**

```
int Kocsi::szekek_szama [private]
```

A kocsi azonosítója.

Ez a dokumentáció az osztályról a következőfájlok alapján készült:

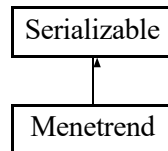
- [kocsi.h](#)
- [kocsi.cpp](#)

## 4.7. Menetrend osztályreferencia

Szabványos bemenet és kimenet kezeléséhez.

```
#include <menetrend.h>
```

A Menetrend osztály származási diagramja:



### Publikus tagfüggvények

- `Menetrend ()`  
*A menetrendben szereplővonatok tömbje.*
- `Menetrend (Vonat v)`
- `Menetrend (Vonat *v, int meret)`
- `Menetrend (const Menetrend &m)`
- `void addVonat (Vonat v)`
- `void changeVonat (Vonat v, int index)`
- `Vonat & getVonat (int index) const`
- `int getVonatokSzama () const`
- `void removeVonat (int index)`
- `void createJegy (std::string indulo, std::string erkezo, int indulo_ora, int indulo_perc, double discountOrFee=0, const std::string &tipus="")`
- `void clear ()`  
*Menetrendben tárolt adatok törlése.*
- `void kiir (std::ostream &os, std::string indulo, std::string erkezo) const`
- `void write (std::ostream &os) const`
- `void read (std::istream &is)`
- `~Menetrend ()`  
*Az osztály destruktora.*

### Privát attribútumok

- `int vonatok_szama`
- `Vonat * vonatok`  
*A menetrendben szereplővonatok száma.*

#### 4.7.1. Részletes leírás

Szabványos bemenet és kimenet kezeléséhez.

Interfész az objektumok sorosításához és deszerializálásához. Memória helyes használatának ellenőrzéséhez. A `Menetrend` osztály reprezentálja a vonatok menetrendjét. Tartalmazza a vonatok számát és a vonatokat.

## 4.7.2. Konstruktorek és destruktorek dokumentációja

### 4.7.2.1. Menetrend() [1/4]

```
Menetrend::Menetrend ( ) [inline]
```

A menetrendben szereplővonatok tömbje.

Az osztály alapértelmezett konstruktora.

### 4.7.2.2. Menetrend() [2/4]

```
Menetrend::Menetrend (
    Vonat v )
```

Paraméteres konstruktork.

#### Paraméterek

|          |                          |
|----------|--------------------------|
| <i>v</i> | - hozzáadni kívánt vonat |
|----------|--------------------------|

### 4.7.2.3. Menetrend() [3/4]

```
Menetrend::Menetrend (
    Vonat v,
    int meret ) [inline]
```

Paraméteres konstruktork.

#### Paraméterek

|              |                              |
|--------------|------------------------------|
| <i>v</i>     | - a vonatok tömbje           |
| <i>meret</i> | - a vonatok tömbjének mérete |

### 4.7.2.4. Menetrend() [4/4]

```
Menetrend::Menetrend (
    const Menetrend & m )
```

Másoló konstruktork.

#### Paraméterek

|          |   |
|----------|---|
| <i>m</i> | - másolni kívánt <a href="#">Menetrend</a> objektum |
|----------|---|

#### 4.7.2.5. ~Menetrend()

```
Menetrend::~~Menetrend ( )
```

Az osztály destruktora.

### 4.7.3. Tagfüggvények dokumentációja

#### 4.7.3.1. addVonat()

```
void Menetrend::addVonat (
    Vonat v )
```

[Vonat](#) hozzáadása a menetrendhez.

##### Paraméterek

|          |                          |
|----------|--------------------------|
| <i>v</i> | - hozzáadni kívánt vonat |
|----------|--------------------------|

#### 4.7.3.2. changeVonat()

```
void Menetrend::changeVonat (
    Vonat v,
    int index )
```

[Vonat](#) módosítása a menetrendben.

##### Paraméterek

|              |                                 |
|--------------|---------------------------------|
| <i>v</i>     | - a módosítandó vonat           |
| <i>index</i> | - a vonat indexe a menetrendben |

#### 4.7.3.3. clear()

```
void Menetrend::clear ( )
```

Menetrendben tárolt adatok törlése.

#### 4.7.3.4. createJegy()

```
void Menetrend::createJegy (
    std::string indulo,
    std::string erkezo,
    int indulo_ora,
    int indulo_perc,
    double discountOrFee = 0,
    const std::string & tipus = "" )
```

[Jegy](#) létrehozása a megadott állomások között.

**Paraméterek**

|                      |                                 |
|----------------------|---------------------------------|
| <i>indulo</i>        | - Az induló állomás neve.       |
| <i>erkezo</i>        | - Az érkezőállomás neve.        |
| <i>indulo_ora</i>    | - Az indulási időórása.         |
| <i>indulo_perc</i>   | - Az indulási időperce.         |
| <i>discountOrFee</i> | - Kedvezmény vagy felár értéke. |
| <i>tipus</i>         | - A jegy típusa.                |

**4.7.3.5. getVonat()**

```
Vonat & Menetrend::getVonat (
    int index ) const
```

*Vonat* lekérdezése a megadott indexen.

**Paraméterek**

|              |                         |
|--------------|-------------------------|
| <i>index</i> | - a kívánt vonat indexe |
|--------------|-------------------------|

**Visszatérési érték**

Az adott indexen található vonat

**4.7.3.6. getVonatokSzama()**

```
int Menetrend::getVonatokSzama ( ) const
```

Vonatok számának lekérdezése.

**Visszatérési érték**

A vonatok száma a menetrendben

**4.7.3.7. kiir()**

```
void Menetrend::kiir (
    std::ostream & os,
    std::string indulo = "",
    std::string erkezo = "" ) const
```

Kiírja a menetrendben található vonatok útvonalát a megadott állomások között. Ha *indulo* és *erkezo* üres, akkor mindet. Ha csak *indulo* üres, akkor kiír minden útvonalat, amely az adott állomásba tart. Ha csak *erkezo* üres, akkor kiír minden útvonalat, amely az adott állomásból indul. Ha *erkezo* és *indulo* üres, akkor kiír minden útvonalat, amely a két állomás között van.

**Paraméterek**

|               |   |
|---------------|---|
| <i>os</i>     | - Az írásra használt kimeneti adatfolyam. |
| <i>indulo</i> | - Az induló állomás neve.                 |
| <i>erkezo</i> | - Az érkezőállomás neve.                  |

**4.7.3.8. read()**

```
void Menetrend::read (
    std::istream & is ) [virtual]
```

[Serializable](#) interfész implementációja: objektum olvasása adatfolyamból.

**Paraméterek**

|           |   |
|-----------|---|
| <i>is</i> | - A beolvasásra használt bemeneti adatfolyam. |
|-----------|---|

Megvalósítja a következőket: [Serializable](#).

**4.7.3.9. removeVonat()**

```
void Menetrend::removeVonat (
    int index )
```

[Vonat](#) törlése a megadott indexen.

**Paraméterek**

|              |                          |
|--------------|--------------------------|
| <i>index</i> | - a törlendővonat indexe |
|--------------|--------------------------|

**4.7.3.10. write()**

```
void Menetrend::write (
    std::ostream & os ) const [virtual]
```

[Serializable](#) interfész implementációja: objektum írása adatfolyamba.

**Paraméterek**

|           |   |
|-----------|---|
| <i>os</i> | - Az írásra használt kimeneti adatfolyam. |
|-----------|---|

Megvalósítja a következőket: [Serializable](#).

## 4.7.4. Adattagok dokumentációja

### 4.7.4.1. vonatok

`Vonat` Menetrend::vonatok [private]

A menetrendben szereplővonatok száma.

### 4.7.4.2. vonatok\_szama

`int` Menetrend::vonatok\_szama [private]

Ez a dokumentáció az osztályról a következőfájlok alapján készült:

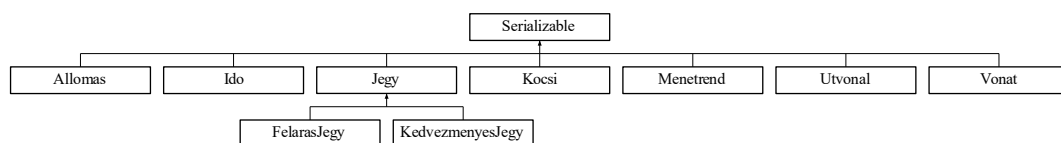
- [menetrend.h](#)
- [menetrend.cpp](#)

## 4.8. Serializable osztályreferencia

Ostream és istream használatához.

```
#include <serializable.h>
```

A Serializable osztály származási diagramja:



### Publikus tagfüggvények

- virtual void `write` (std::ostream &os) const =0
- virtual void `read` (std::istream &is)=0

### 4.8.1. Részletes leírás

Ostream és istream használatához.

Memória helyes használatának ellenőrzéséhez. Absztrakt osztály a perzisztencia megvalósításához. Tartalmazza a `write()` és `read()` virtuális függvényeket, amelyeket a leszármazott osztályoknak meg kell valósítaniuk.

### 4.8.2. Tagfüggvények dokumentációja

#### 4.8.2.1. read()

```
virtual void Serializable::read (
    std::istream & is ) [pure virtual]
```

Adatfolyamról olvasásért felelős függvény.



**Paraméterek**

|                 |   |
|-----------------|---|
| <code>is</code> | - Az objektum olvasása során használt bemeneti adatfolyam, például egy fájl vagy a konzol bemenete. |
|-----------------|---|

Megvalósítják a következők: [Allomas](#), [Ido](#), [Jegy](#), [KedvezmenyesJegy](#), [FelarasJegy](#), [Kocsi](#), [Menetrend](#), [Utvonal](#) és [Vonat](#).

**4.8.2.2. write()**

```
virtual void Serializable::write (
    std::ostream & os ) const [pure virtual]
```

Adatfolyamra írásért felelős függvény.

**Paraméterek**

|                 |  |
|-----------------|--|
| <code>os</code> | - Az objektum írása során használt kimeneti adatfolyam, például egy fájl vagy a konzol kimenete. |
|-----------------|--|

Megvalósítják a következők: [Allomas](#), [Ido](#), [Jegy](#), [KedvezmenyesJegy](#), [FelarasJegy](#), [Kocsi](#), [Menetrend](#), [Utvonal](#) és [Vonat](#).

Ez a dokumentáció az osztályról a következőfájl alapján készült:

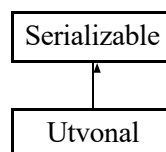
- [serializable.h](#)

**4.9. Utvonal osztályreferencia**

Interfész az objektumok sorosításához és deszerializálásához.

```
#include <utvonal.h>
```

Az Utvonal osztály származási diagramja:

**Publikus tagfüggvények**

- [Utvonal](#) ()  
*Az állomások tömbje az útvonalon.*
- [Utvonal](#) (const [Utvonal](#) &other)
- void [addAllomas](#) ([Allomas](#) allomas)
- void [createAllomas](#) (std::string nev, int erkezes\_ora, int erkezes\_perc, int indulas\_ora, int indulas\_perc)
- [Utvonal](#) & [operator=](#) (const [Utvonal](#) &other)
- void [changeAllomas](#) (int index, [Allomas](#) allomas)

- void `removeAllomas` (int index)
- int `getAllomasokSzama` () const  
*Az útvonalon található állomások számának lekérdezése.*
- `Allomas & getAllomas` (int i) const
- void `kiir` (std::ostream &os) const
- void `write` (std::ostream &os) const
- void `read` (std::istream &is)
- `~Utvonal` ()  
*Az osztály destruktora.*

### Privát attribútumok

- int `utvonal_azonosito`
- int `allomasok_szama`  
*Az útvonal azonosítója.*
- `Allomas * allomasok`  
*Az állomások száma az útvonalon.*

## 4.9.1. Részletes leírás

Interfész az objektumok sorosításához és deszerializálásához.

Memória helyes használatának ellenőrzéséhez. Az `Utvonal` osztály az egyes vonatútvonalakat reprezentálja. Tartalmazza az útvonal azonosítóját, az állomások számát, valamint az állomásokat.

## 4.9.2. Konstruktorok és destruktorok dokumentációja

### 4.9.2.1. `Utvonal()` [1/2]

```
Utvonal::Utvonal ( )
```

Az állomások tömbje az útvonalon.

Az osztály alapértelmezett konstruktora.

### 4.9.2.2. `Utvonal()` [2/2]

```
Utvonal::Utvonal (
    const Utvonal & other )
```

Az osztály másoló konstruktora.

### Paraméterek

|                    |  |
|--------------------|--|
| <code>other</code> | - másolni kívánt <code>Utvonal</code> objektum |
|--------------------|--|

#### 4.9.2.3. ~Útvonal()

```
Útvonal::~~Útvonal ( )
```

Az osztály destruktora.

### 4.9.3. Tagfüggvények dokumentációja

#### 4.9.3.1. addAllomas()

```
void Útvonal::addAllomas (
    Allomas allomas )
```

Állomás hozzáadása az útvonalhoz.

##### Paraméterek

|                |                            |
|----------------|----------------------------|
| <i>allomas</i> | - hozzáadni kívánt állomás |
|----------------|----------------------------|

#### 4.9.3.2. changeAllomas()

```
void Útvonal::changeAllomas (
    int index,
    Allomas allomas )
```

Állomás cseréje az útvonalon.

##### Paraméterek

|                |  |
|----------------|--|
| <i>index</i>   | - az állomás indexe, amelyet cserélni kívánunk |
| <i>allomas</i> | - beszúrni kívánt állomás                      |

#### 4.9.3.3. createAllomas()

```
void Útvonal::createAllomas (
    std::string nev,
    int erkezes_ora,
    int erkezes_perc,
    int indulas_ora,
    int indulas_perc )
```

Új állomás létrehozása az útvonalon.

##### Paraméterek

|                     |                                   |
|---------------------|-----------------------------------|
| <i>nev</i>          | - az állomás neve                 |
| <i>erkezes_ora</i>  | - az állomásra érkezővonat óra    |
| <i>erkezes_perc</i> | - az állomásra érkezővonat perc   |
| <i>indulas_ora</i>  | - az állomásról induló vonat óra  |
| <i>indulas_perc</i> | - az állomásról induló vonat perc |

#### 4.9.3.4. getAllomas()

```
Allomas & Utvonal::getAllomas (
    int i ) const
```

Visszaad egy kívánt számú állomást a listából az index alapján.

##### Paraméterek

|          |                     |
|----------|---------------------|
| <i>i</i> | - az állomás indexe |
|----------|---------------------|

##### Visszatérési érték

Az állomás referenciája az adott indexen

#### 4.9.3.5. getAllomasokSzama()

```
int Utvonal::getAllomasokSzama ( ) const
```

Az útvonalon található állomások számának lekérdezése.

#### 4.9.3.6. kiir()

```
void Utvonal::kiir (
    std::ostream & os ) const
```

Objektum kiírása adatfolyamba esztétikus formában.

##### Paraméterek

|           |   |
|-----------|---|
| <i>os</i> | - Az írásra használt kimeneti adatfolyam. |
|-----------|---|

#### 4.9.3.7. operator=()

```
Utvonal & Utvonal::operator= (
    const Utvonal & other )
```

Az útvonal értékadás operátora.

##### Paraméterek

|              |   |
|--------------|---|
| <i>other</i> | - másolni kívánt <a href="#">Utvonal</a> objektum |
|--------------|---|

#### 4.9.3.8. read()

```
void Utvonal::read (
```

```
std::istream & is ) [virtual]
```

[Serializable](#) interfész implementációja: objektum olvasása adatfolyamból.

#### Paraméterek

|           |   |
|-----------|---|
| <i>is</i> | - A beolvasásra használt bemeneti adatfolyam. |
|-----------|---|

Megvalósítja a következőket: [Serializable](#).

#### 4.9.3.9. removeAllomas()

```
void Utvonal::removeAllomas (
    int index )
```

Állomás eltávolítása az útvonalról.

#### Paraméterek

|              |   |
|--------------|---|
| <i>index</i> | - az eltávolítani kívánt állomás indexe |
|--------------|---|

#### 4.9.3.10. write()

```
void Utvonal::write (
    std::ostream & os ) const [virtual]
```

[Serializable](#) interfész implementációja: objektum írása adatfolyamba.

#### Paraméterek

|           |   |
|-----------|---|
| <i>os</i> | - Az írásra használt kimeneti adatfolyam. |
|-----------|---|

Megvalósítja a következőket: [Serializable](#).

### 4.9.4. Adattagok dokumentációja

#### 4.9.4.1. allomasok

```
Allomas& Utvonal::allomasok [private]
```

Az állomások száma az útvonalon.

#### 4.9.4.2. allomasok\_szama

```
int Utvonal::allomasok_szama [private]
```

Az útvonal azonosítója.

#### 4.9.4.3. utvonal\_azonosito

```
int Utvonal::utvonal_azonosito [private]
```

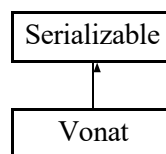
Ez a dokumentáció az osztályról a következőfájlok alapján készült:

- [utvonal.h](#)
- [utvonal.cpp](#)

## 4.10. Vonat osztályreferencia

```
#include <vonat.h>
```

A Vonat osztály származási diagramja:



### Publikus tagfüggvények

- [Vonat \(\)](#)  
*A vonathoz tartozó útvonala.*
- [Vonat \(const Vonat &other\)](#)
- [Vonat & operator= \(const Vonat &other\)](#)
- [Vonat \(int azonosito, int kocsi\\_sz, \[Kocsi\]\(#\) kocsi\\_tomb\[\], \[Utvonal\]\(#\) utv, int jegyek\\_sz, \[Jegy\]\(#\) \\*\\*jegyek\\_ptr\)](#)
- [int getAzonosito \(\) const](#)
- [Utvonal getUtvonal \(\) const](#)
- [void setAzonosito \(int azonosito\)](#)
- [void setUtvonal \(const Utvonal &utvonal\)](#)
- [void addKocsi \(Kocsi &kocsi\)](#)
- [void createKocsi \(int szekek\)](#)
- [void addJegy \(Jegy &jegy\)](#)
- [Jegy \\* getJegy \(int index\) const](#)
- [bool routeExists \(std::string indulo, std::string erkezo\) const](#)
- [int findAllomas \(std::string nev\) const](#)
- [int indulasidoKulonbseg \(std::string nev, int ora, int perc\)](#)
- [int createJegy \(std::string indulo, std::string erkezo, int indulo\\_ora=0, int indulo\\_perc=0, double discountOr↵  
Fee=0, const std::string &tipus=""\)](#)
- [void write \(std::ostream &os\) const](#)
- [void read \(std::istream &is\)](#)
- [~Vonat \(\)](#)  
*Az osztály destruktora.*

**Privát attribútumok**

- int `vonat_azonosito`
- int `kocsik_szama`  
*A vonat azonosítója.*
- int `jegyek_szama`  
*A vonathoz tartozó kocsik száma.*
- `Kocsi * kocsik`  
*A vonathoz tartozó jegyek száma.*
- `Jegy ** jegyek`  
*A vonathoz tartozó kocsik tömbje.*
- `Utvonal utvonal`  
*A vonathoz tartozó jegyek tömbje.*

**4.10.1. Részletes leírás**

A `Vonat` osztály a reprezentálja a vonatokat. Tartalmazza a vonat azonosítóját, a vonathoz tartozó kocsik számát, a kocsik tömbjét, a vonathoz tartozó jegyeket, azok számát, valamint az útvonalát.

**4.10.2. Konstruktork és destruktorok dokumentációja****4.10.2.1. Vonat() [1/3]**

```
Vonat::Vonat ( )
```

A vonathoz tartozó útvonal.

Az osztály alapértelmezett konstruktora.

**4.10.2.2. Vonat() [2/3]**

```
Vonat::Vonat (
    const Vonat & other )
```

Az osztály másoló konstruktora.

**Paraméterek**

|                    |   |
|--------------------|---|
| <code>other</code> | - Másolni kívánt <code>Vonat</code> objektum. |
|--------------------|---|

**4.10.2.3. Vonat() [3/3]**

```
Vonat::Vonat (
    int azonosito,
    int kocsik_sz,
    Kocsi kocsik_tomb[],
```

```
Utvonal utv,  
int jegyek_sz,  
Jegy ** jegyek_ptr )
```

Paraméteres konstruktor.

#### Paraméterek

|                    |  |
|--------------------|--|
| <i>azonosito</i>   | - A vonat azonosítója.                   |
| <i>kocsik_sz</i>   | - A vonathoz tartozó kocsik száma.       |
| <i>kocsik_tomb</i> | - A kocsik tömbje.                       |
| <i>utv</i>         | - Az útvonal, amelyhez a vonat tartozik. |
| <i>jegyek_sz</i>   | - A vonathoz tartozó jegyek száma.       |
| <i>jegyek_ptr</i>  | - A jegyek tömbjére mutató pointer.      |

#### 4.10.2.4. ~Vonat()

```
Vonat::~~Vonat ( )
```

Az osztály destruktora.

### 4.10.3. Tagfüggvények dokumentációja

#### 4.10.3.1. addJegy()

```
void Vonat::addJegy (  
    Jegy & jegy )
```

Jegy hozzáadása a vonathoz.

#### Paraméterek

|             |                         |
|-------------|-------------------------|
| <i>jegy</i> | - hozzáadni kívánt jegy |
|-------------|-------------------------|

#### 4.10.3.2. addKocsi()

```
void Vonat::addKocsi (  
    Kocsi & kocsi )
```

Kocsi hozzáadása a vonathoz.

#### Paraméterek

|              |                          |
|--------------|--------------------------|
| <i>kocsi</i> | - hozzáadni kívánt kocsi |
|--------------|--------------------------|



#### 4.10.3.3. createJegy()

```
int Vonat::createJegy (
    std::string indulo,
    std::string erkezo,
    int indulo_ora = 0,
    int indulo_perc = 0,
    double discountOrFee = 0,
    const std::string & tipus = "" )
```

Új jegy létrehozása az adott indulási és érkezési állomások között.

##### Paraméterek

|                      |   |
|----------------------|---|
| <i>indulo</i>        | - Az indulási állomás neve.   |
| <i>erkezo</i>        | - Az érkezési állomás neve.   |
| <i>indulo_ora</i>    | - Az indulási időórája (alapértelmezett érték: 0).                  |
| <i>indulo_perc</i>   | - Az indulási időperce (alapértelmezett érték: 0).                  |
| <i>discountOrFee</i> | - Kedvezmény vagy felár mértéke (alapértelmezett érték: 0).         |
| <i>tipus</i>         | - <a href="#">Jegy</a> típusa (alapértelmezett érték: üres string). |

##### Visszatérési érték

Az újonnan létrehozott jegy azonosítója.

#### 4.10.3.4. createKocsi()

```
void Vonat::createKocsi (
    int szekek )
```

Új kocsi létrehozása a vonathoz.

##### Paraméterek

|               |                                      |
|---------------|--------------------------------------|
| <i>szekek</i> | - Az új kocsin elérhetőszékek száma. |
|---------------|--------------------------------------|

#### 4.10.3.5. findAllomas()

```
int Vonat::findAllomas (
    std::string nev ) const
```

Megkeresi az állomás indexét az állomás neve alapján.

##### Paraméterek

|            |  |
|------------|--|
| <i>nev</i> | - Az állomás neve, aminek az indexét keresi. |
|------------|--|

**Visszatérési érték**

Az állomás indexe, vagy -1, ha nem található az állomás.

**4.10.3.6. getAzonosito()**

```
int Vonat::getAzonosito ( ) const
```

Azonosító lekérdezése.

**Visszatérési érték**

A vonat azonosítója.

**4.10.3.7. getJegy()**

```
Jegy * Vonat::getJegy (
    int index ) const
```

Jegy lekérdezése adott indexen.

**Paraméterek**

|              |                            |
|--------------|----------------------------|
| <i>index</i> | - A jegy indexe a tömbben. |
|--------------|----------------------------|

**Visszatérési érték**

Az adott indexen található jegy pointer.

**4.10.3.8. getUtvonal()**

```
Utvonal Vonat::getUtvonal ( ) const
```

Útvonal lekérdezése.

**Visszatérési érték**

Az útvonal, amelyhez a vonat tartozik.

**4.10.3.9. indulasiIdoKulonbseg()**

```
int Vonat::indulasiIdoKulonbseg (
    std::string nev,
    int ora,
    int perc )
```

Meghatározza, hogy hány perc különbség van a megadott időpont és a vonat indulásának időpontja között.

**Paraméterek**

|             |   |
|-------------|---|
| <i>nev</i>  | - Az állomás neve, ahonnan az indulási időt számítja. |
| <i>ora</i>  | - Az indulási időórája.                               |
| <i>perc</i> | - Az indulási időperce.                               |

**Visszatérési érték**

Perc különbség van a megadott időpont és a vonat indulásának időpontja között vagy -1, ha az állomás nem található vagy a vonat hamarabb indul mint a megadott időpont.

**4.10.3.10. operator=()**

```
Vonat & Vonat::operator= (
    const Vonat & other )
```

Értékadó operátor.

**Paraméterek**

|              |  |
|--------------|--|
| <i>other</i> | - Másolni kívánt <a href="#">Vonat</a> objektum. |
|--------------|--|

**Visszatérési érték**

Az értékadás eredménye, az új [Vonat](#) objektum referenciája.

**4.10.3.11. read()**

```
void Vonat::read (
    std::istream & is ) [virtual]
```

[Serializable](#) interfész implementációja: objektum olvasása adatfolyamból.

**Paraméterek**

|           |   |
|-----------|---|
| <i>is</i> | - A beolvasásra használt bemeneti adatfolyam. |
|-----------|---|

Megvalósítja a következőket: [Serializable](#).

**4.10.3.12. routeExists()**

```
bool Vonat::routeExists (
    std::string indulo,
    std::string erkezo ) const
```

Ellenőrzi, hogy létezik-e útvonal az adott indulási és érkezési állomások között.

**Paraméterek**

|               |                             |
|---------------|-----------------------------|
| <i>indulo</i> | - Az indulási állomás neve. |
| <i>erkezo</i> | - Az érkezési állomás neve. |

**Visszatérési érték**

igaz, ha létezik útvonal, különben hamis.

**4.10.3.13. setAzonosito()**

```
void Vonat::setAzonosito (
    int azonosito )
```

Azonosító beállítása.

**Paraméterek**

|                  |                            |
|------------------|----------------------------|
| <i>azonosito</i> | - Az új vonat azonosítója. |
|------------------|----------------------------|

**4.10.3.14. setUtvonal()**

```
void Vonat::setUtvonal (
    const Utvonal & utvonal )
```

Útvonal beállítása.

**Paraméterek**

|                |   |
|----------------|---|
| <i>utvonal</i> | - az útvonal, amelyhez a vonat tartozik |
|----------------|---|

**4.10.3.15. write()**

```
void Vonat::write (
    std::ostream & os ) const [virtual]
```

[Serializable](#) interfész implementációja: objektum írása adatfolyamba.

**Paraméterek**

|           |   |
|-----------|---|
| <i>os</i> | - Az írásra használt kimeneti adatfolyam. |
|-----------|---|

Megvalósítja a következőket: [Serializable](#).

#### 4.10.4. Adattagok dokumentációja

##### 4.10.4.1. jegyek

`Jegy` Vonat::jegyek [private]

A vonathoz tartozó kocsik tömbje.

##### 4.10.4.2. jegyek\_szama

`int` Vonat::jegyek\_szama [private]

A vonathoz tartozó kocsik száma.

##### 4.10.4.3. kocsik

`Kocsi` Vonat::kocsik [private]

A vonathoz tartozó jegyek száma.

##### 4.10.4.4. kocsik\_szama

`int` Vonat::kocsik\_szama [private]

A vonat azonosítója.

##### 4.10.4.5. utvonal

`Utvonal` Vonat::utvonal [private]

A vonathoz tartozó jegyek tömbje.

##### 4.10.4.6. vonat\_azonosito

`int` Vonat::vonat\_azonosito [private]

Ez a dokumentáció az osztályról a következőfájlok alapján készült:

- [vonat.h](#)
- [vonat.cpp](#)



## 5. fejezet

# Fájlok dokumentációja

### 5.1. allomas.cpp fájlreferencia

```
#include "allomas.h"  
#include <cstring>  
#include <cstdint>  
#include <iostream>
```

### 5.2. allomas.h fájlreferencia

```
#include "ido.h"  
#include <iostream>  
#include "memtrace.h"  
#include "serializable.h"
```

#### Osztályok

- class [Allomas](#)  
*Időtárolásához.*

### 5.3. allomas.h

[Ugrás a fájl dokumentációjához.](#)

```
00001 #ifndef ALLOMAS_H  
00002 #define ALLOMAS_H  
00003  
00005 #include "ido.h"  
00006  
00008 #include <iostream>  
00009  
00011 #include "memtrace.h"  
00012  
00014 #include "serializable.h"  
00015  
00017 class Allomas : public Serializable  
00018 {
```

```
00019     int allomas_azonosito;
00020     std::string nev;
00021     Ido indulas;
00022     Ido erkezes;
00023 public:
00024     Allomas();
00025
00026     Allomas(const Allomas &other);
00027
00028     Allomas &operator=(const Allomas &other);
00029
00030     Allomas(int azonosito, const std::string &nev, int erkezes_ora, int erkezes_perc, int indulas_ora,
00031             int indulas_perc);
00032
00033     void changeErkezes(int ora, int perc);
00034
00035     void changeIndulas(int ora, int perc);
00036
00037     void changeNev(const std::string &other);
00038
00039     int getIndulasOra();
00040
00041     int getIndulasPerc();
00042
00043     int getErkezesOra();
00044
00045     int getErkezesPerc();
00046
00047     Ido &getIndulas();
00048
00049     Ido &getErkezes();
00050
00051     std::string &getNev();
00052
00053     void write(std::ostream &os) const;
00054
00055     void read(std::istream &is);
00056
00057     ~Allomas();
00058 };
00059 #endif
```

## 5.4. ido.cpp fájlreferencia

```
#include "ido.h"
```

## 5.5. ido.h fájlreferencia

```
#include <iostream>
#include "serializable.h"
#include "memtrace.h"
```

### Osztályok

- class `Ido`

*Szabványos bemenet és kimenet kezeléséhez.*



## 5.6. ido.h

[Ugrás a fájl dokumentációjához.](#)

```
00001 #ifndef IDO_H
00002 #define IDO_H
00003
00005 #include <iostream>
00006
00008 #include "serializable.h"
00009
00011 #include "memtrace.h"
00012
00015 class Ido : public Serializable
00016 {
00017 private:
00018     int ora;
00019     int perc;
00020 public:
00022     Ido();
00023
00027     Ido(int o, int p);
00028
00031     Ido(const Ido &other);
00032
00035     int getOra() const;
00036
00039     int getPerc() const;
00040
00043     void setOra(int o);
00044
00047     void setPerc(int p);
00048
00052     void setIdo(int o, int p);
00053
00056     void addPerc(int p);
00057
00060     void addOra(int o);
00061
00065     bool operator==(const Ido &other) const;
00066
00070     bool operator!=(const Ido &other) const;
00071
00074     void kiir(std::ostream &os) const;
00075
00078     void write(std::ostream &os) const;
00079
00082     void read(std::istream &is);
00083
00085     ~Ido();
00086 };
00087
00088 #endif
```

## 5.7. jegy.cpp fájlreferencia

```
#include "jegy.h"
#include <iostream>
```

## 5.8. jegy.h fájlreferencia

```
#include <iostream>
#include "serializable.h"
#include "memtrace.h"
#include "allomas.h"
#include "ido.h"
```

## Osztályok

- class [Jegy](#)  
Szabványos bemenet és kimenet kezeléséhez.
- class [KedvezmenyesJegy](#)
- class [FelarasJegy](#)

## 5.9. jegy.h

[Ugrás a fájl dokumentációjához.](#)

```
00001 #ifndef JEGY_H
00002 #define JEGY_H
00003
00005 #include <iostream>
00006
00008 #include "serializable.h"
00009
00011 #include "memtrace.h"
00012
00013 #include "allomas.h"
00014
00016 #include "ido.h"
00017
00021 class Jegy : public Serializable
00022 {
00023 protected:
00024     int jegy_azonosito;
00025     int helyszam;
00026     int kocsi_szam;
00027     int vonat_szam;
00028     std::string indulo_nev;
00029     Idó indulo_ido;
00030     std::string cel_nev;
00031     Idó cel_ido;
00032     int ar;
00033     virtual int getAr(double pred);
00034 public:
00036     Jegy();
00037
00045     Jegy(int azonosito, int hely, int kocsi, int vonat, const std::string &indulo, Idó indulo_idopont,
00046         const std::string &cel, Idó cel_idopont);
00047
00056     Jegy(int azonosito, int hely, int kocsi, int vonat, const std::string &indulo, Idó indulo_idopont,
00057         const std::string &cel, Idó cel_idopont, double pred);
00058
00061     Jegy(const Jegy &other);
00062
00066     Jegy &operator=(const Jegy &other);
00067
00071     bool operator==(const Jegy &other) const;
00072
00076     bool operator!=(const Jegy &other) const;
00077
00080     void kiir(std::ostream &os) const;
00081
00084     void write(std::ostream &os) const;
00085
00088     void read(std::istream &is);
00089
00092     virtual Jegy *clone() const;
00093
00095     virtual ~Jegy() {}
00096 };
00097
00100 class KedvezmenyesJegy : public Jegy
00101 {
00102     std::string tipus;
00103     double kedvezmeny;
00104 public:
00106     KedvezmenyesJegy();
00107
00110     KedvezmenyesJegy(const KedvezmenyesJegy &other);
00111
00121     KedvezmenyesJegy(int azonosito, int hely, int kocsi, int vonat, const std::string &indulo, Idó
00122 indulo_idopont,
00123         const std::string &cel, Idó cel_idopont, double kedvezmeny, std::string tipus);
00127     KedvezmenyesJegy &operator=(const KedvezmenyesJegy &other);
```

```

00128
00131     void kiir(std::ostream &os) const;
00132
00135     void write(std::ostream &os) const;
00136
00139     void read(std::istream &is);
00140
00143     Jegy *clone() const;
00144
00146     ~KedvezmenyesJegy();
00147 };
00148
00151 class FelarasJegy : public Jegy
00152 {
00153     std::string tipus;
00154     double felar;
00155 public:
00157     FelarasJegy();
00158
00161     FelarasJegy(const FelarasJegy &other);
00162
00172     FelarasJegy(int azonosito, int hely, int kocsi, int vonat, const std::string &indulo, Idó
indulo_idopont,
00173                 const std::string &cel, Idó cel_idopont, double felar, std::string tipus);
00174
00178     FelarasJegy &operator=(const FelarasJegy &other);
00179
00182     void kiir(std::ostream &os) const;
00183
00186     void write(std::ostream &os) const;
00187
00190     void read(std::istream &is);
00191
00194     Jegy *clone() const;
00195
00197     ~FelarasJegy();
00198 };
00199
00200 #endif

```

## 5.10. kocsi.cpp fájlreferencia

```
#include "kocsi.h"
```

## 5.11. kocsi.h fájlreferencia

```

#include <iostream>
#include "serializable.h"
#include "memtrace.h"

```

### Osztályok

- class [Kocsi](#)

*Szabványos bemenet és kimenet kezeléséhez.*

## 5.12. kocsi.h

[Ugrás a fájl dokumentációjához.](#)

```
00001 #ifndef KOCST_H
00002 #define KOCST_H
00003
00005 #include <iostream>
00006
00008 #include "serializable.h"
00009
00011 #include "memtrace.h"
00012
00015 class Kocsi : public Serializable
00016 {
00017     int kocsi_azonosito;
00018     int szekek_szama;
00019     int foglalt_szekek_szama;
00020 public:
00022     Kocsi();
00023
00026     Kocsi(const Kocsi &other);
00027
00031     Kocsi(int azonosito, int szekek);
00032
00035     bool tele() const;
00036
00039     int reserveHely();
00040
00043     int getSzekek_szama() const;
00044
00047     int getAzonosito() const;
00048
00051     void write(std::ostream &os) const;
00052
00055     void read(std::istream &is);
00056
00058     ~Kocsi();
00059 };
00060
00061 #endif
```

## 5.13. main.cpp fájlreferencia

```
#include "menetrend.h"
#include "vonat.h"
#include "jegy.h"
#include "utvonal.h"
#include "ido.h"
#include "allomas.h"
#include "kocsi.h"
#include "test.h"
#include <iostream>
#include <sstream>
#include <fstream>
#include "memtrace.h"
```

### Függvények

- int [main](#) ()

### 5.13.1. Függvények dokumentációja

#### 5.13.1.1. main()

```
int main ( )
```

## 5.14. menetrend.cpp fájlreferencia

```
#include "menetrend.h"
```

## 5.15. menetrend.h fájlreferencia

```
#include "vonat.h"  
#include <iostream>  
#include "serializable.h"  
#include "memtrace.h"
```

### Osztályok

- class [Menetrend](#)

*Szabványos bemenet és kimenet kezeléséhez.*

## 5.16. menetrend.h

[Ugrás a fájl dokumentációjához.](#)

```
00001 #ifndef MENETREND_H  
00002 #define MENETREND_H  
00003  
00004 #include "vonat.h"  
00005  
00007 #include <iostream>  
00008  
00010 #include "serializable.h"  
00011  
00013 #include "memtrace.h"  
00014  
00017 class Menetrend : public Serializable  
00018 {  
00019     int vonatok_szama;  
00020     Vonat *vonatok;  
00021 public:  
00023     Menetrend() : vonatok_szama(0), vonatok(NULL) {}  
00024  
00027     Menetrend(Vonat v);  
00028  
00032     Menetrend(Vonat *v, int meret) : vonatok_szama(meret), vonatok(v) {}  
00033  
00036     Menetrend(const Menetrend &m);  
00037  
00040     void addVonat(Vonat v);  
00041  
00045     void changeVonat(Vonat v, int index);  
00046  
00050     Vonat &getVonat(int index) const;  
00051  
00054     int getVonatokSzama() const;  
00055  
00058     void removeVonat(int index);  
00059  
00067     void createJegy(std::string indulo, std::string erkezo, int indulo_ora, int indulo_perc, double  
00068         discountOrFee = 0, const std::string &tipus = "");  
00070     void clear();  
00071  
00080     void kiir(std::ostream &os, std::string indulo, std::string erkezo) const;  
00081  
00084     void write(std::ostream &os) const;  
00085  
00088     void read(std::istream &is);  
00089  
00091     ~Menetrend();  
00092 };  
00093  
00094 #endif
```

## 5.17. serializable.h fájlreferencia

```
#include <iostream>
#include "memtrace.h"
```

### Osztályok

- class [Serializable](#)

*Ostream és istream használatához.*

## 5.18. serializable.h

[Ugrás a fájl dokumentációjához.](#)

```
00001 #ifndef SERIALIZABLE_H
00002 #define SERIALIZABLE_H
00003
00005 #include <iostream>
00006
00008 #include "memtrace.h"
00009
00012 class Serializable
00013 {
00014 public:
00017     virtual void write(std::ostream &os) const = 0;
00018
00021     virtual void read(std::istream &is) = 0;
00022 };
00023
00024 #endif
```

## 5.19. test.cpp fájlreferencia

```
#include "test.h"
```

### Függvények

- void [test](#) ()

*Teszteléskor vizsgált osztályok.*

### 5.19.1. Függvények dokumentációja

#### 5.19.1.1. test()

```
void test ( )
```

Teszteléskor vizsgált osztályok.

Programhoz tartozó tesztek futtató függvény

**Paraméterek**

|                   |  |
|-------------------|--|
| <code>void</code> |  |
|-------------------|--|

**Visszatérési  
érték**`void`

Időosztály kiírás funkciójának vizsgálata

StringStream létrehozása a kiíráshoz

időobjektumok létrehozása paraméteres konstruktorokkal 10

óra, 12 perc

2 óra, 1 perc

időobjektumok kiírása vart:

"10:12"

vart: "02:01"

kiírt értékek vizsgálata

/// Időosztály perzisztenciájának vizsgálata

StringStream létrehozása

időobjektum létrehozása paraméteres konstruktorokkal 8

óra, 30 perc

időobjektum létrehozása beolvasáshoz

ido1 objektum kiírása StringStream-be

ido2 objektum beolvasása a StringStream-ből

időobjektumok egyezésének vizsgálata

[Ido](#) osztály alapvető műveleteinek vizsgálata

időobjektum létrehozása paraméteres konstruktorokkal 2

óra, 45 perc

konstruktor értékbeállításának vizsgálata

új értékek beállítása setter függvényekkel

setterek működésekének ellenőrzése

órák és percek hozzáadása

aritmetikai művelet ellenőrzése 3 óra 15 perc + 2 óra 30 perc = 5 óra 45perc

időobjektum létrehozása egyezés vizsgálathoz

egyenlőség operátor ellenőrzése

különbözőség operátor ellenőrzése

Állomás konstruktorának vizsgálata

állomás létrehozása konstruktorral

paraméterek helyes beállítottságának vizsgálata

állomás létrehozása paraméteres konstruktorral

paraméterek helyes beállítottságának vizsgálata

Állomás osztály attribútum változtatásának vizsgálata

állomás létrehozása paraméteres konstruktorral

indulás és érkezés időpontjainak változtatása

indulás és érkezés sikeres változtatásának ellenőrzése

állomás létrehozása paraméteres konstruktorral

nev paraméter módosítása

nev paraméter állításának ellenőrzése

Állomás osztály perzisztenciájának vizsgálata

StringStream létrehozása

állomás létrehozása paraméteres konstruktorral

állomás létrehozása beolvasáshoz

állomás kiírása StringStream-be

állomás beolvasása StringStream-ből

eredeti es beolvasott objektum egyezésének vizsgálata

Útvonal osztály állomás hozzáadás függvényének vizsgálata

Útvonal objektum létrehozása

állomás objektum létrehozása paraméteres konstruktorral

állomások hozzáadása az útvonalhoz

útvonalban tárolt állomások ellenőrzése

Útvonal állomás cseréjének vizsgálata

Útvonal létrehozása

állomások létrehozása paraméteres konstruktorral



állomások hozzáadása az útvonalhoz

útvonalban tárolt 2. állomás cseréje másik állomásra

útvonalban tárolt állomások vizsgálata a csere után

Útvonal osztály perzisztenciájának vizsgálata

Útvonal létrehozása

állomások létrehozása paraméteres konstruktorral

állomások hozzáadása az útvonalhoz

StringStream létrehozása

útvonal objektum kiírása StringStream-be

útvonal objektum létrehozása beolvasáshoz

útvonal objektum beolvasása StringStream-ből

útvonal objektumok egyezésének vizsgálata

Kocsi osztály szék foglalás funkciójának vizsgálata

Kocsi objektum létrehozása 5 u" löhely kapacitással

minden u" löhely lefoglalása

nincs több szabad hely, foglalásnál hibát kell dobnia

Kocsi osztály foglaltság ellenőrzőfüggvényének vizsgálata

Kocsi objektum létrehozása 2 u" löhely kapacitással

van szabad hely, a tele függvény hamis értéket kell adjon

minden u" löhely lefoglalása

nincs több szabad hely, a tele függvény igaz értéket kell adjon

Kocsi osztály perzisztenciájának vizsgálata

Kocsi objektum létrehozása 3 u" löhely kapacitással

1 hely lefoglalása

StringStream létrehozása

kocsi objektum kiírása StringStream-be

kocsi objektum létrehozása beolvasáshoz

kocsi objektum beolvasása a StringStream-ből

kocsi objektumok egyezésének vizsgálata

Vonat osztály útvonal beállító és lekérdezőfüggvényeinek vizsgálata

Útvonal objektum létrehozása

állomások létrehozása az útvonalon

[Vonat](#) objektum létrehozása

útvonal beállítása

útvonal lekérdezése és eltárolása

eredeti es beolvasott objektum egyezésének vizsgálata

[Vonat](#) osztály jegy hozzáadó függvényének vizsgálata

[Vonat](#) objektumok létrehozása

időobjektum létrehozása paraméteres konstruktorokkal 8

óra, 0 perc

10 óra, 30 perc

[Jegy](#) létrehozása

jegy hozzáadása a vonathoz

eredeti és vonat-ból kapott jegyek egyezésének vizsgálata

[Vonat](#) osztály indulási időkülönbség számító függvényének vizsgálata Útvonal

objektum létrehozása

állomások létrehozása az útvonalon

[Vonat](#) objektum létrehozása

útvonal beállítása

időkülönbség kiszámítása a vonat indulása Budapest állomásról és a megadott időpont között

különbség ellenőrzése az állomásról való indulás ideje 10:30, ami 21 percel később van mint a megadott 10:09

[Vonat](#) osztály perzisztenciájának vizsgálata

Kocsik létrehozása

Időobjektumok létrehozása Jegyek

létrehozása

Útvonal létrehozása és állomások hozzáadása

állomások létrehozása az útvonalon

StringStream létrehozása

vonat objektum kiírása StringStream-be

vonat objektum létrehozása beolvasáshoz

kcosi objektum beolvasása a StringStream-ből

vonatok egyezésének vizsgálata

[Jegy](#) osztály == operátorának vizsgálata

időobjektumok létrehozása paraméteres konstruktorokkal

jegyek létrehozása

jegyek egyezésének vizsgálata

[Jegy](#) osztály másoló konstruktorának vizsgálata

időobjektumok létrehozása paraméteres konstruktorokkal

jegy létrehozása

jegy létrehozása másoló konstruktorral

jegyek egyezésének vizsgálata

[Jegy](#) osztály értékadó operátorának vizsgálata

időobjektumok létrehozása paraméteres konstruktorokkal

jegy létrehozása

jegy létrehozása értékadáshoz

jegy értéküladása

jegyek egyezésének vizsgálata

[Jegy](#) osztály perzisztenciájának vizsgálata

időobjektumok létrehozása paraméteres konstruktorokkal

jegy létrehozása

stringstream létrehozása

jegy kiírása stringstream-be

jegy létrehozása beolvasáshoz

jegy beolvasása stringstream-ből

jegyek egyezésének vizsgálata

Felarasjegy osztály perzisztenciájának vizsgálata

időobjektumok létrehozása paraméteres konstruktorokkal

felarasjegy létrehozása

stringstream létrehozása

jegy kiírása stringstream-be

jegy létrehozása beolvasáshoz  
jegy beolvasása StringStream-ből  
jegyek egyezésének vizsgálata  
Kedvezményesjegy osztály perzisztenciájának vizsgálata  
időobjektumok létrehozása paraméteres konstruktorokkal  
kedvezményesjegy létrehozása  
StringStream létrehozása  
jegy kiírása StringStream-be  
jegy létrehozása beolvasáshoz  
jegy beolvasása StringStream-ből  
jegyek egyezésének vizsgálata  
Állomás konstruktorának vizsgálata  
menetrend objektum létrehozása  
paraméterek helyes beállítottságának vizsgálata  
[Menetrend](#) vonat hozzáadó függvényének vizsgálata  
menetrend objektum létrehozása  
útvonal objektum létrehozása  
állomások létrehozása az útvonalon  
vonat létrehozása az útvonallal  
vonat hozzáadása a menetrendhez  
vonat hozzáadásának ellenőrzése  
[Menetrend](#) vonat módosításának vizsgálata  
menetrend objektum létrehozása  
útvonal objektumok létrehozása  
állomások létrehozása az útvonalon  
vonatok létrehozása az útvonallal  
vonat hozzáadása a menetrendhez  
hozzáadott vonat cseréje  
vonat cseréjének ellenőrzése  
[Menetrend](#) osztály perzisztenciájának vizsgálata  
menetrend objektum létrehozása  
útvonal objektum létrehozása  
állomások létrehozása az útvonalon  
vonat létrehozása az útvonallal  
vonat hozzáadása a menetrendhez  
[Menetrend](#) kiírása StringStream-be  
menetrend kiírása StringStream-be  
[Menetrend](#) objektum létrehozása beolvasáshoz  
menetrend beolvasása StringStream-ből  
menetrendek egyezésének vizsgálata

## 5.20. test.h fájlreferencia

```
#include "menetrend.h"
#include "vonat.h"
#include "jegy.h"
#include "utvonal.h"
#include "ido.h"
#include "allomas.h"
#include "kocsi.h"
#include <sstream>
#include <iostream>
#include "memtrace.h"
#include "gtest_lite.h"
```

### Függvények

- void `test` ()

*Teszteléskor vizsgált osztályok.*

### 5.20.1. Függvények dokumentációja

#### 5.20.1.1. test()

```
void test ( )
```

Teszteléskor vizsgált osztályok.

Perzisztencia vizsgálatához. Memória helyes használatának ellenőrzéséhez. Teszteléshez használt makrók. Programhoz tartozó tesztek futtató függvény.

Programhoz tartozó tesztek futtató függvény

#### Paraméterek

|                   |  |
|-------------------|--|
| <code>void</code> |  |
|-------------------|--|

#### Visszatérési érték

void

Időosztály kiírás funkciójának vizsgálata

StringStream létrehozása a kiíráshoz

időobjektumok létrehozása paraméteres konstruktorokkal 10

óra, 12 perc

2 óra, 1 perc

időobjektumok kiírása vart:

"10:12"

vart: "02:01"

kiírt értékek vizsgálata

/// Időosztály perzisztenciájának vizsgálata

StringStream létrehozása

időobjektum létrehozása paraméteres konstruktorokkal 8

óra, 30 perc

időobjektum létrehozása beolvasáshoz

ido1 objektum kiírása StringStream-be

ido2 objektum beolvasása a StringStream-ből

időobjektumok egyezésének vizsgálata

[Ido](#) osztály alapvető műveleteinek vizsgálata

időobjektum létrehozása paraméteres konstruktorokkal 2

óra, 45 perc

konstruktor értékbeállításának vizsgálata

új értékek beállítása setter függvényekkel

setterek működésekének ellenőrzése

órák és percek hozzáadása

aritmetikai művelet ellenőrzése 3 óra 15 perc + 2 óra 30 perc = 5 óra 45perc

időobjektum létrehozása egyezés vizsgálathoz

egyenlőség operátor ellenőrzése

különbözőség operátor ellenőrzése

Állomás konstruktorának vizsgálata

állomás létrehozása konstruktorral

paraméterek helyes beállítottságának vizsgálata

állomás létrehozása paraméteres konstruktorral

paraméterek helyes beállítottságának vizsgálata

Állomás osztály attribútum változtatásának vizsgálata

állomás létrehozása paraméteres konstruktorral

indulás és érkezés időpontjainak változtatása

indulás és érkezés sikeres változtatásának ellenőrzése

állomás létrehozása paraméteres konstruktorral

nev paraméter módosítása

nev paraméter állításának ellenőrzése

Állomás osztály perzisztenciájának vizsgálata

StringStream létrehozása

állomás létrehozása paraméteres konstruktorral

állomás létrehozása beolvasáshoz

állomás kiírása StringStream-be

állomás beolvasása StringStream-ből

eredeti es beolvasott objektum egyezésének vizsgálata

Útvonal osztály állomás hozzáadás függvényének vizsgálata

Útvonal objektum létrehozása

állomás objektum létrehozása paraméteres konstruktorral

állomások hozzáadása az útvonalhoz

útvonalban tárolt állomások ellenőrzése

Útvonal állomás cseréjének vizsgálata

Útvonal létrehozása

állomások létrehozása paraméteres konstruktorral

állomások hozzáadása az útvonalhoz

útvonalban tárolt 2. állomás cseréje másik állomásra

útvonalban tárolt állomások vizsgálata a csere után

Útvonal osztály perzisztenciájának vizsgálata

Útvonal létrehozása

állomások létrehozása paraméteres konstruktorral

állomások hozzáadása az útvonalhoz

StringStream létrehozása

útvonal objektum kiírása StringStream-be

útvonal objektum létrehozása beolvasáshoz

útvonal objektum beolvasása StringStream-ből

útvonal objektumok egyezésének vizsgálata

[Kocsi](#) osztály szék foglalás funkciójának vizsgálata

[Kocsi](#) objektum létrehozása 5 u" lőhely kapacitással

minden u" lőhely lefoglalása

nincs több szabad hely, foglalásnál hibát kell dobnia

[Kocsi](#) osztály foglaltság ellenőrzőfüggvényének vizsgálata

[Kocsi](#) objektum létrehozása 2 u" lőhely kapacitással

van szabad hely, a tele függvény hamis értéket kell adjon

minden u" lőhely lefoglalása

nincs több szabad hely, a tele függvény igaz értéket kell adjon

[Kocsi](#) osztály perzisztenciájának vizsgálata

[Kocsi](#) objektum létrehozása 3 u" lőhely kapacitással

1 hely lefoglalása

StringStream létrehozása

kocsi objektum kiírása StringStream-be

kocsi objektum létrehozása beolvasáshoz

kcosi objektum beolvasása a StringStream-ből

kocsi objektumok egyezésének vizsgálata

[Vonat](#) osztály útvonal beállító és lekérdezőfüggvényeinek vizsgálata Útvonal

objektum létrehozása

állomások létrehozása az útvonalon

[Vonat](#) objektum létrehozása

útvonal beállítása

útvonal lekérdezése és eltárolása

eredeti es beolvasott objektum egyezésének vizsgálata

[Vonat](#) osztály jegy hozzáadó függvényének vizsgálata

[Vonat](#) objektumok létrehozása

időobjektum létrehozása paraméteres konstruktorokkal 8

óra, 0 perc



10 óra, 30 perc

[Jegy](#) létrehozása

jegy hozzáadása a vonathoz

eredeti és vonat-ból kapott jegyek egyezésének vizsgálata

[Vonat](#) osztály indulási időkülönbség számító függvényének vizsgálata Útvonal

objektum létrehozása

állomások létrehozása az útvonalon

[Vonat](#) objektum létrehozása

útvonal beállítása

időkülönbség kiszámítása a vonat indulása Budapest állomásról és a megadott időpont között

különbség ellenőrzése az állomásról való indulás ideje 10:30, ami 21 percel később van mint a megadott 10:09

[Vonat](#) osztály perzisztenciájának vizsgálata

Kocsik létrehozása

Időobjektumok létrehozása Jegyek

létrehozása

Útvonal létrehozása és állomások hozzáadása

állomások létrehozása az útvonalon

StringStream létrehozása

vonat objektum kiírása StringStream-be

vonat objektum létrehozása beolvasáshoz

kcosi objektum beolvasása a StringStream-ből

vonatok egyezésének vizsgálata

[Jegy](#) osztály == operátorának vizsgálata

időobjektumok létrehozása paraméteres konstruktorokkal

jegyek létrehozása

jegyek egyezésének vizsgálata

[Jegy](#) osztály másoló konstruktorának vizsgálata

időobjektumok létrehozása paraméteres konstruktorokkal

jegy létrehozása

jegy létrehozása másoló konstruktorral

jegyek egyezésének vizsgálata

[Jegy](#) osztály értékadó operátorának vizsgálata

időobjektumok létrehozása paraméteres konstruktorokkal

jegy létrehozása

jegy létrehozása értékadáshoz

jegy értéküladása

jegyek egyezésének vizsgálata

[Jegy](#) osztály perzisztenciájának vizsgálata

időobjektumok létrehozása paraméteres konstruktorokkal

jegy létrehozása

StringStream létrehozása

jegy kiírása StringStream-be

jegy létrehozása beolvasáshoz

jegy beolvasása StringStream-ből

jegyek egyezésének vizsgálata

Felarasjegy osztály perzisztenciájának vizsgálata

időobjektumok létrehozása paraméteres konstruktorokkal

felarasjegy létrehozása

StringStream létrehozása

jegy kiírása StringStream-be

jegy létrehozása beolvasáshoz

jegy beolvasása StringStream-ből

jegyek egyezésének vizsgálata

Kedvezményesjegy osztály perzisztenciájának vizsgálata

időobjektumok létrehozása paraméteres konstruktorokkal

kedvezményesjegy létrehozása

StringStream létrehozása

jegy kiírása StringStream-be

jegy létrehozása beolvasáshoz

jegy beolvasása StringStream-ből

jegyek egyezésének vizsgálata

Állomás konstruktorának vizsgálata

menetrend objektum létrehozása

paraméterek helyes beállítottságának vizsgálata

[Menetrend](#) vonat hozzáadó függvényének vizsgálata

menetrend objektum létrehozása

útvonal objektum létrehozása

állomások létrehozása az útvonalon

vonat létrehozása az útvonallal

vonat hozzáadása a menetrendhez

vonat hozzáadásának ellenőrzése

[Menetrend](#) vonat módosításának vizsgálata

menetrend objektum létrehozása

útvonal objektumok létrehozása

állomások létrehozása az útvonalon

vonatok létrehozása az útvonallal

vonat hozzáadása a menetrendhez

hozzáadott vonat cseréje

vonat cseréjének ellenőrzése

[Menetrend](#) osztály perzisztenciájának vizsgálata

menetrend objektum létrehozása

útvonal objektum létrehozása

állomások létrehozása az útvonalon

vonat létrehozása az útvonallal

vonat hozzáadása a menetrendhez

[Menetrend](#) kiírása StringStream-be

menetrend kiírása StringStream-be

[Menetrend](#) objektum létrehozása beolvasáshoz

menetrend beolvasása StringStream-ből

menetrendek egyezésének vizsgálata

## 5.21. test.h

[Ugrás a fájl dokumentációjához.](#)

```
00001 #ifndef TEST_H_
00002 #define TEST_H_
00003
00005 #include "menetrend.h"
00006 #include "vonat.h"
00007 #include "jegy.h"
00008 #include "utvonal.h"
00009 #include "ido.h"
00010 #include "allomas.h"
00011 #include "kocsi.h"
00012
00014 #include <sstream>
00015 #include <iostream>
00016
00018 #include "memtrace.h"
00020 #include "gtest_lite.h"
00021
00023 void test();
00024
00025 #endif /* TEST_H_ */
```

## 5.22. utvonal.cpp fájlreferencia

```
#include "utvonal.h"
```

## 5.23. utvonal.h fájlreferencia

```
#include "allomas.h"
#include "serializable.h"
#include "memtrace.h"
```

### Osztályok

- class [Utvonal](#)

*Interfész az objektumok sorosításához és deszerializálásához.*

## 5.24. utvonal.h

[Ugrás a fájl dokumentációjához.](#)

```
00001 #ifndef UTVONAL_H
00002 #define UTVONAL_H
00003
00004 #include "allomas.h"
00005
00007 #include "serializable.h"
00008
00010 #include "memtrace.h"
00011
00014 class Utvonal : public Serializable
00015 {
00016     int utvonal_azonosito;
00017     int allomasok_szama;
00018     Allomas *allomasok;
00019 public:
00021     Utvonal();
```

```

00022
00025     Utvonal(const Utvonal &other);
00026
00029     void addAllomas(Allomas allomas);
00030
00037     void createAllomas(std::string nev, int erkezes_ora, int erkezes_perc, int indulas_ora, int
indulas_perc);
00038
00041     Utvonal &operator=(const Utvonal &other);
00042
00046     void changeAllomas(int index, Allomas allomas);
00047
00050     void removeAllomas(int index);
00051
00053     int getAllomasokSzama() const;
00054
00058     Allomas &getAllomas(int i) const;
00059
00062     void kiir(std::ostream &os) const;
00063
00066     void write(std::ostream &os) const;
00067
00070     void read(std::istream &is);
00071
00073     ~Utvonal();
00074 };
00075
00076 #endif

```

## 5.25. vonat.cpp fájlreferencia

```
#include "vonat.h"
```

## 5.26. vonat.h fájlreferencia

```

#include <iostream>
#include "memtrace.h"
#include "serializable.h"
#include "kocsi.h"
#include "jegy.h"
#include "utvonal.h"

```

### Osztályok

- class [Vonat](#)

## 5.27. vonat.h

[Ugrás a fájl dokumentációjához.](#)

```

00001 #ifndef VONAT_H
00002 #define VONAT_H
00003
00004 #include <iostream>
00005
00006 #include "memtrace.h"
00007
00008 #include "serializable.h"
00009 #include "kocsi.h"
00010 #include "jegy.h"
00011 #include "utvonal.h"

```

```
00012
00016 class Vonat : public Serializable
00017 {
00018 private:
00019     int vonat_azonosito;
00020     int kocsik_szama;
00021     int jegyek_szama;
00022     Kocsi *kocsik;
00023     Jegy **jegyek;
00024     Utvonal utvonal;
00025 public:
00027     Vonat();
00028
00031     Vonat(const Vonat &other);
00032
00036     Vonat &operator=(const Vonat &other);
00037
00045     Vonat(int azonosito, int kocsik_sz, Kocsi kocsik_tomb[], Utvonal utv, int jegyek_sz, Jegy
**jegyek_ptr);
00046
00049     int getAzonosito() const;
00050
00053     Utvonal getUtvonal() const;
00054
00057     void setAzonosito(int azonosito);
00058
00061     void setUtvonal(const Utvonal &utvonal);
00062
00065     void addKocsi(Kocsi &kocsi);
00066
00069     void createKocsi(int szekek);
00070
00073     void addJegy(Jegy &jegy);
00074
00078     Jegy *getJegy(int index) const;
00079
00084     bool routeExists(std::string indulo, std::string erkezo) const;
00085
00089     int findAllomas(std::string nev) const;
00090
00097     int indulasiIdoKulonbseg(std::string nev, int ora, int perc);
00098
00107     int createJegy(std::string indulo, std::string erkezo, int indulo_ora = 0, int indulo_perc = 0,
double discountOrFee = 0, const std::string &tipus = "");
00108
00111     void write(std::ostream &os) const;
00112
00115     void read(std::istream &is);
00116
00118     ~Vonat();
00119 };
00120
00121 #endif
```

# Tárgymutató

- ~Allomas
  - Allomas, [10](#)
- ~FelarasJegy
  - FelarasJegy, [16](#)
- ~Ido
  - Ido, [19](#)
- ~Jegy
  - Jegy, [27](#)
- ~KedvezmenyesJegy
  - KedvezmenyesJegy, [33](#)
- ~Kocsi
  - Kocsi, [37](#)
- ~Menetrend
  - Menetrend, [40](#)
- ~Utvonal
  - Utvonal, [46](#)
- ~Vonat
  - Vonat, [52](#)
- addAllomas
  - Utvonal, [47](#)
- addJegy
  - Vonat, [52](#)
- addKocsi
  - Vonat, [52](#)
- addOra
  - Ido, [20](#)
- addPerc
  - Ido, [20](#)
- addVonat
  - Menetrend, [41](#)
- Allomas, [7](#)
  - ~Allomas, [10](#)
  - Allomas, [8](#)
  - allomas\_azonosito, [13](#)
  - changeErkezes, [10](#)
  - changeIndulas, [10](#)
  - changeNev, [10](#)
  - erkezes, [13](#)
  - getErkezes, [11](#)
  - getErkezesOra, [11](#)
  - getErkezesPerc, [11](#)
  - getIndulas, [11](#)
  - getIndulasOra, [11](#)
  - getIndulasPerc, [12](#)
  - getNev, [12](#)
  - indulas, [13](#)
  - nev, [13](#)
  - operator=, [12](#)
  - read, [12](#)
  - write, [13](#)
- allomas.cpp, [59](#)
- allomas.h, [59](#)
- allomas\_azonosito
  - Allomas, [13](#)
- allomasok
  - Utvonal, [49](#)
- allomasok\_szama
  - Utvonal, [49](#)
- ar
  - Jegy, [29](#)
- cel\_ido
  - Jegy, [29](#)
- cel\_nev
  - Jegy, [29](#)
- changeAllomas
  - Utvonal, [47](#)
- changeErkezes
  - Allomas, [10](#)
- changeIndulas
  - Allomas, [10](#)
- changeNev
  - Allomas, [10](#)
- changeVonat
  - Menetrend, [41](#)
- clear
  - Menetrend, [41](#)
- clone
  - FelarasJegy, [16](#)
  - Jegy, [27](#)
  - KedvezmenyesJegy, [33](#)
- createAllomas
  - Utvonal, [47](#)
- createJegy
  - Menetrend, [41](#)
  - Vonat, [52](#)
- createKocsi
  - Vonat, [53](#)
- erkezes
  - Allomas, [13](#)
- felar
  - FelarasJegy, [18](#)
- FelarasJegy, [14](#)
  - ~FelarasJegy, [16](#)
  - clone, [16](#)
  - felar, [18](#)

- FelarasJegy, 15, 16
- kiir, 16
- operator=, 17
- read, 17
- tipus, 18
- write, 17
- findAllomas
  - Vonat, 53
- foglalt\_szekek\_szama
  - Kocsi, 38
- getAllomas
  - Utvonal, 48
- getAllomasokSzama
  - Utvonal, 48
- getAr
  - Jegy, 27
- getAzonosito
  - Kocsi, 37
  - Vonat, 54
- getErkezes
  - Allomas, 11
- getErkezesOra
  - Allomas, 11
- getErkezesPerc
  - Allomas, 11
- getIndulas
  - Allomas, 11
- getIndulasOra
  - Allomas, 11
- getIndulasPerc
  - Allomas, 12
- getJegy
  - Vonat, 54
- getNev
  - Allomas, 12
- getOra
  - Ido, 20
- getPerc
  - Ido, 20
- getSzekek\_szama
  - Kocsi, 37
- getUtvonal
  - Vonat, 54
- getVonat
  - Menetrend, 42
- getVonatokSzama
  - Menetrend, 42
- helyszam
  - Jegy, 30
- Ido, 18
  - ~Ido, 19
  - addOra, 20
  - addPerc, 20
  - getOra, 20
  - getPerc, 20
  - Ido, 19
  - kiir, 20
  - operator!=, 21
  - operator==, 21
  - ora, 24
  - perc, 24
  - read, 21
  - setIdo, 22
  - setOra, 22
  - setPerc, 22
  - write, 22
- ido.cpp, 60
- ido.h, 60
- indulas
  - Allomas, 13
- indulasidoKulonbseg
  - Vonat, 54
- indulo\_ido
  - Jegy, 30
- indulo\_nev
  - Jegy, 30
- Jegy, 24
  - ~Jegy, 27
  - ar, 29
  - cel\_ido, 29
  - cel\_nev, 29
  - clone, 27
  - getAr, 27
  - helyszam, 30
  - indulo\_ido, 30
  - indulo\_nev, 30
  - Jegy, 26, 27
  - jegy\_azonosito, 30
  - kiir, 27
  - kocsi\_szam, 30
  - operator!=, 28
  - operator=, 28
  - operator==, 28
  - read, 29
  - vonat\_szam, 30
  - write, 29
- jegy.cpp, 61
- jegy.h, 61
- jegy\_azonosito
  - Jegy, 30
- jegyek
  - Vonat, 57
- jegyek\_szama
  - Vonat, 57
- kedvezmeny
  - KedvezmenyesJegy, 35
- KedvezmenyesJegy, 31
  - ~KedvezmenyesJegy, 33
  - clone, 33
  - kedvezmeny, 35
  - KedvezmenyesJegy, 32, 33
  - kiir, 33
  - operator=, 34



- read, [34](#)
- tipus, [35](#)
- write, [34](#)
- kiír
  - FelarasJegy, [16](#)
  - Ido, [20](#)
  - Jegy, [27](#)
  - KedvezmenyesJegy, [33](#)
  - Menetrend, [42](#)
  - Utvonal, [48](#)
- Kocsi, [35](#)
  - ~Kocsi, [37](#)
  - foglalt\_szekek\_szama, [38](#)
  - getAzonosito, [37](#)
  - getSzekek\_szama, [37](#)
  - Kocsi, [36](#)
  - kocsi\_azonosito, [38](#)
  - read, [37](#)
  - reserveHely, [37](#)
  - szekek\_szama, [38](#)
  - tele, [38](#)
  - write, [38](#)
- kocsi.cpp, [63](#)
- kocsi.h, [63](#)
- kocsi\_azonosito
  - Kocsi, [38](#)
- kocsi\_szam
  - Jegy, [30](#)
- kocsik
  - Vonat, [57](#)
- kocsik\_szama
  - Vonat, [57](#)
- main
  - main.cpp, [64](#)
- main.cpp, [64](#)
  - main, [64](#)
- Menetrend, [39](#)
  - ~Menetrend, [40](#)
  - addVonat, [41](#)
  - changeVonat, [41](#)
  - clear, [41](#)
  - createJegy, [41](#)
  - getVonat, [42](#)
  - getVonatokSzama, [42](#)
  - kiír, [42](#)
  - Menetrend, [40](#)
  - read, [43](#)
  - removeVonat, [43](#)
  - vonatok, [44](#)
  - vonatok\_szama, [44](#)
  - write, [43](#)
- menetrend.cpp, [65](#)
- menetrend.h, [65](#)
- nev
  - Allomas, [13](#)
- operator!=
  - Ido, [21](#)
  - Jegy, [28](#)
- operator=
  - Allomas, [12](#)
  - FelarasJegy, [17](#)
  - Jegy, [28](#)
  - KedvezmenyesJegy, [34](#)
  - Utvonal, [48](#)
  - Vonat, [55](#)
- operator==
  - Ido, [21](#)
  - Jegy, [28](#)
- ora
  - Ido, [24](#)
- perc
  - Ido, [24](#)
- read
  - Allomas, [12](#)
  - FelarasJegy, [17](#)
  - Ido, [21](#)
  - Jegy, [29](#)
  - KedvezmenyesJegy, [34](#)
  - Kocsi, [37](#)
  - Menetrend, [43](#)
  - Serializable, [44](#)
  - Utvonal, [48](#)
  - Vonat, [55](#)
- removeAllomas
  - Utvonal, [49](#)
- removeVonat
  - Menetrend, [43](#)
- reserveHely
  - Kocsi, [37](#)
- routeExists
  - Vonat, [55](#)
- Serializable, [44](#)
  - read, [44](#)
  - write, [45](#)
- serializable.h, [66](#)
- setAzonosito
  - Vonat, [56](#)
- setIdo
  - Ido, [22](#)
- setOra
  - Ido, [22](#)
- setPerc
  - Ido, [22](#)
- setUtvonal
  - Vonat, [56](#)
- szekek\_szama
  - Kocsi, [38](#)
- tele
  - Kocsi, [38](#)
- test
  - test.cpp, [66](#)

test.h, [73](#)  
test.cpp, [66](#)  
    test, [66](#)  
test.h, [73](#)  
    test, [73](#)  
tipus  
    FelarasJegy, [18](#)  
    KedvezmenyesJegy, [35](#)  
  
Utvonal, [45](#)  
    ~Utvonal, [46](#)  
    addAllomas, [47](#)  
    allomasok, [49](#)  
    allomasok\_szama, [49](#)  
    changeAllomas, [47](#)  
    createAllomas, [47](#)  
    getAllomas, [48](#)  
    getAllomasokSzama, [48](#)  
    kiir, [48](#)  
    operator=, [48](#)  
    read, [48](#)  
    removeAllomas, [49](#)  
    Utvonal, [46](#)  
    utvonal\_azonosito, [49](#)  
    write, [49](#)  
utvonal  
    Vonat, [57](#)  
utvonal.cpp, [80](#)  
utvonal.h, [80](#)  
utvonal\_azonosito  
    Utvonal, [49](#)  
  
Vonat, [50](#)  
    ~Vonat, [52](#)  
    addJegy, [52](#)  
    addKocsi, [52](#)  
    createJegy, [52](#)  
    createKocsi, [53](#)  
    findAllomas, [53](#)  
    getAzonosito, [54](#)  
    getJegy, [54](#)  
    getUtvonal, [54](#)  
    indulasidoKulonbseg, [54](#)  
    jegyek, [57](#)  
    jegyek\_szama, [57](#)  
    kocsik, [57](#)  
    kocsik\_szama, [57](#)  
    operator=, [55](#)  
    read, [55](#)  
    routeExists, [55](#)  
    setAzonosito, [56](#)  
    setUtvonal, [56](#)  
    utvonal, [57](#)  
    Vonat, [51](#)  
    vonat\_azonosito, [57](#)  
    write, [56](#)  
vonat.cpp, [81](#)  
vonat.h, [81](#)  
vonat\_azonosito  
    Vonat, [57](#)  
    vonat\_szam  
        Jegy, [30](#)  
vonatok  
    Menetrend, [44](#)  
vonatok\_szama  
    Menetrend, [44](#)  
  
write  
    Allomas, [13](#)  
    FelarasJegy, [17](#)  
    Ido, [22](#)  
    Jegy, [29](#)  
    KedvezmenyesJegy, [34](#)  
    Kocsi, [38](#)  
    Menetrend, [43](#)  
    Serializable, [45](#)  
    Utvonal, [49](#)  
    Vonat, [56](#)