

# A PROGRAMOZÁS ALAPJAI 2.

## HÁZI FELADAT DOKUMENTÁCIÓ

### Vonat menetrend

KÉSZÍTETTE: WITTMANN MÁTYÁS, OXHER7  
[wittmann.matyas@gmail.com](mailto:wittmann.matyas@gmail.com)

KÉSZÍTÉS FÉLÉVE: 2023/22/2

# TARTALOMJEGYZÉK

Felhasználói dokumentáció .....	3
Osztályok statikus leírása .....	4
RailwayNetwork .....	4
Felelőssége .....	4
Attribútumok .....	4
Metódusok .....	4
Train .....	5
Felelőssége .....	5
Attribútumok .....	5
Metódusok .....	5
TimeTable .....	6
Felelőssége .....	6
Attribútumok .....	6
Metódusok .....	6
TrainTime .....	7
Felelőssége .....	7
Attribútumok .....	7
Metódusok .....	7
UML osztálydiagramm .....	8
Összegzés .....	9
Mit sikerült és mit nem sikerült megvalósítani a specifikációból? .....	9
Mit tanultál a megvalósítás során? .....	9
Továbbfejlesztési lehetőségek .....	9
Képernyőképek a futó alkalmazásról .....	10

# Felhasználói dokumentáció

A program paraméternek kap 2 filenevet pl.: ./~/Nagy\ Házi\ Feladat/Házi\ Feladat/bin/Debug/Házi\ Feladat allomasok.csv vonatok.csv . A két file a program gyökérkönyvtárban található. Amennyiben a program filet hoz létre azokat szintén itt találjuk. A program feltétele a helyes megjelenítéséhez egy olyan shell és terminál/standard output kombináció ami tud unicode karaktereket kezelni (én konsole 24.02.2-t használtam ami tud unicode karaktereket kezelni).

Amennyiben sikeresen elindítottuk a programot egy menü segítségével navigálhatunk a program funkciói között. A menü strktúra az alábbi:

- 1: kiíratás a terminálba
- 2: kiíratás fileba
- 3: vonat menetidő lekérdezés
- 0: kilépés

A program ezután vár a felhasználotól egy számot 0-3 ig, a program ezután az alábbi menüpontok alapján jár el.

Az 1-es menü a standard outputra írja ki a menetrendet, valamennyire formázott, táblázatszerű módon. Amennyiben végeztünk a táblázat megtekintésével egy enter lenyomása után visszatérhetünk a főmenübe. Egy példa a program kimenetére:

```
menetrend

| id | Jelenlegi állomás | Érkezések |
-----|-----|-----|
| ERHFD | Makkoshotyka | Apacatorna: 03:39 Bugyi: 05:15 Pornoapati: 07:28 Peneszlek: 12:03 Budapest: 14:28 |
| OTBVE | Apacatorna | Bugyi: 01:55 Pornoapati: 04:35 Peneszlek: 10:06 Budapest: 13:00 |
| KOLOG | Bugyi | Peneszlek: 03:11 Budapest: 06:05 Apacatorna: 07:45 |
| DRTVB | Peneszlek | Budapest: 02:04 Makkoshotyka: 03:51 Apacatorna: 06:59 Bugyi: 08:21 Pornoapati: 10:15 |
| KJITG | Budapest | Makkoshotyka: 01:15 Apacatorna: 03:27 Bugyi: 04:24 Pornoapati: 05:44 |
| FZHDJ | Pornoapati | Peneszlek: 04:31 Makkoshotyka: 05:41 Apacatorna: 09:17 Bugyi: 10:51 |
| UHFGI | Makkoshotyka | Bugyi: 02:12 Pornoapati: 04:19 Peneszlek: 08:41 Budapest: 10:59 |
| TZFGP | Apacatorna | Bugyi: 01:28 Pornoapati: 03:31 Budapest: 05:21 Makkoshotyka: 07:16 |
| ZGHJD | Bugyi | Pornoapati: 02:03 Peneszlek: 06:17 Budapest: 08:31 Makkoshotyka: 10:26 Apacatorna: 13:49 |
| HTFGD | Budapest | Bugyi: 00:20 Pornoapati: 02:31 |

Nyomj enter-t a folytatáshoz !
```

A 2-es menü a vonatok menetrendjét fileba írja, és az állomások egymástól való idejét írja ki, nem az abszolút menetidőt (a terminálos kiírással ellentétben). A program kér a felhasználotól egy kiterjesztés nélküli filenevet, ehhez fűzi hozzá majd a csv kiterjesztést. Amennyiben sikeres a file írás azt a program közli és egy enter lenyomásával visszánavigálhatunk a főmenübe.

```
menetrend

Mit szeretnél csinálni?
1: kiíratás a terminálba
2: kiíratás fileba
3: vonat menetidő lekérdezés
0: kilépés
2
Írd be a file nevét: test
A file írás sikeres! (test.csv)

Nyomj enter-t a folytatáshoz !
```

A 3-as menü két állomás között számolja ki egy adott vonat menetidejét. A menetidőt a megadott vonat sebességéből számolja. A felhasználotól kéri két állomás nevét, vagy space-el vagy enter-el elválasztva, majd a vonat 5 betűs azonosítóját, itt is szintén az enter billentyű lenyomásával térhetünk vissza.

A 0-ás menü segítségével kiléphetünk a programból.

# Osztályok statikus leírása

## RailwayNetwork

### Felelőssége

Egy vasúti hálózatot kezel, vasútállomásokkal, meg tudja mondani két áll.

### Attribútumok

#### Privát

- `string* stations` -> ez egy string tömb, az állomások nevei vannak benne
- `unsigned stationsNum` -> a stations elemszáma
- `map<string, map<string, int>> table` -> a távolságokat tartalmazó 2D map

### Metódusok

#### Publikus

- `RailwayNetwork()` -> default konstruktor
- `~RailwayNetwork()` -> destruktork
- `map<string, map<string, int>> getTable() const` -> a table tagváltozó gettere
- `string& getStation(const unsigned index) const` -> a stations index-ik tagjával tér vissza
- `unsigned getStationsNum () const` -> az állomások számával tér vissza
- `int getTableCell(const string& from, const string& to) const` -> a map egy adott elemével tér vissza
- `void setStations(const string& stationFrom, const string& stationTo, const unsigned newData)` -> a map egy celláját lehet beállítani
- `void initTable()` -> a map főátlót kinullázza
- `void addStation(const string& newData)` -> egy állomást fűz a stations-hoz
- `void loadTableFromCSV(const string& filename)` -> egy csv file-ból betölti a map-et és az állomásokat
- `int getWordLength() const` -> visszatér az állomások szavainak szóhosszáinak összegével

# Train

## Felelőssége

Egy vasúti hálózatot kezel, vasútállomásokkal, meg tudja mondani két áll.

## Attribútumok

### Privát

- string id -> a vonat azonosítója
- string currentStation -> az állomás amin a vonat épp áll
- string\* stations -> az állomások amiken a vonat megáll
- int statNum -> stations elemszáma
- int speed -> a vonat sebessége

## Metódusok

### Publikus

- Train() -> default konstruktor
- Train(string id\_, string currentStation\_, int speed\_, string\* stations\_, int statNum\_) -> konstruktor
- Train(const Train& other) -> másoló konstruktor
- Train(const Train\* other) -> másoló konstruktor
- ~Train() -> destruktork
- void setId(string id\_) -> az ID-tag settere
- void setCurrentStation(string currentStation\_) -> a jelenlegi állomás settere
- void setSpeed(int speed\_) -> a Speed settere
- void addStations(const string& newData) -> állomásokat tud hozzáfűzni a vonathoz (stations)
- string getId() const -> visszatér az ID taggal
- string getCurrentStation() const -> A jelenlegi állomással tér vissza
- int getSpeed() const -> a vonat sebességével tér vissza
- string\* getStations() const -> visszatér a vonat állomásaival (stations tömb)
- int getStatNum() const -> az állomások számával tér vissza
- string& operator[] -> egy operátor függvény ami a stations-ból adja vissza az index-ik elemet
- void generateID() -> generál a vonatnak egy random 5 nagybetűből álló ID-t
- bool isAtStation(const string& station) const -> visszatér True/False, hogy a vonat az állomáson van-e
- double calculateTime(int distance) const -> kiszámolja egy távolságból és a vonat sebességéből a menetidőt
- void printTrain() const -> kiírja egy vonat ID és Speed tagjait
- void printStations() const -> kiírja a vonat állomásait

# TimeTable

## Felelőssége

Ez végzi a menetrend előállítását, kezelését, ez köti össze a Train és railwayNewtwork classokat. (a TimeTable és train között kompozíciós kapcsolat van, a TimeTable és a railwayNewtwork között pedig aggregációs kapcsolat áll fent).

## Attribútumok

### Privát

- Train\*\* trains -> trains-ből álló tömb (kompozíció)
- int trainNum -> a trains dinamikus tag elemszáma
- RailwayNetwork& railway -> az a vasúthálózat amiből nézzük a menetrendet (aggregációs)
- TrainTime start -> az első vonat indulási ideje

## Metódusok

### Publikus

- TimeTable() -> default konstruktor
- TimeTable(RailwayNetwork& railway\_) -> a referencia van megadva, konstruktor
- TimeTable(Train\*\* trains, int trainNum, RailwayNetwork& railway, TrainTime start) -> teljes konsturktor
- ~TimeTable() -> destruktork
- Train\*\* getTrains() const -> visszatér a trains tömbbel
- int getTrainNum() const -> a vonatok számával tér vissza
- RailwayNetwork& getRailway() const -> a railway referenciával tér vissza
- TrainTime getStart() const -> az indulási idővel tér vissza
- Train\* getTrain (const string& id) const -> egy vonatot keres ki a trains tömbből id alapján
- void setTrains(Train\*\* trains\_, int trainNum\_) -> beállítja a trains tömböt
- void setStart(TrainTime start\_) -> az indulási időt állítja be
- void addTrain(const Train\* train) -> hozzáfűz a trains-hez egy vonatot
- void loadTrainsFromFile(const string& filename) -> a vonatok tömböt egy csv file-ból olvassa be
- void saveTimeTable(const string& filename) const -> elmenti a menetrendet egy csv fileba
- void printTimeTable() const -> a standard outputra írja ki a menetrendet

# TrainTime

## Felelőssége

A vonatokkal kapcsolatos idők kezelése, időket ad össze, végez rajtuk logikai műveleteket.

## Attribútumok

### Privát

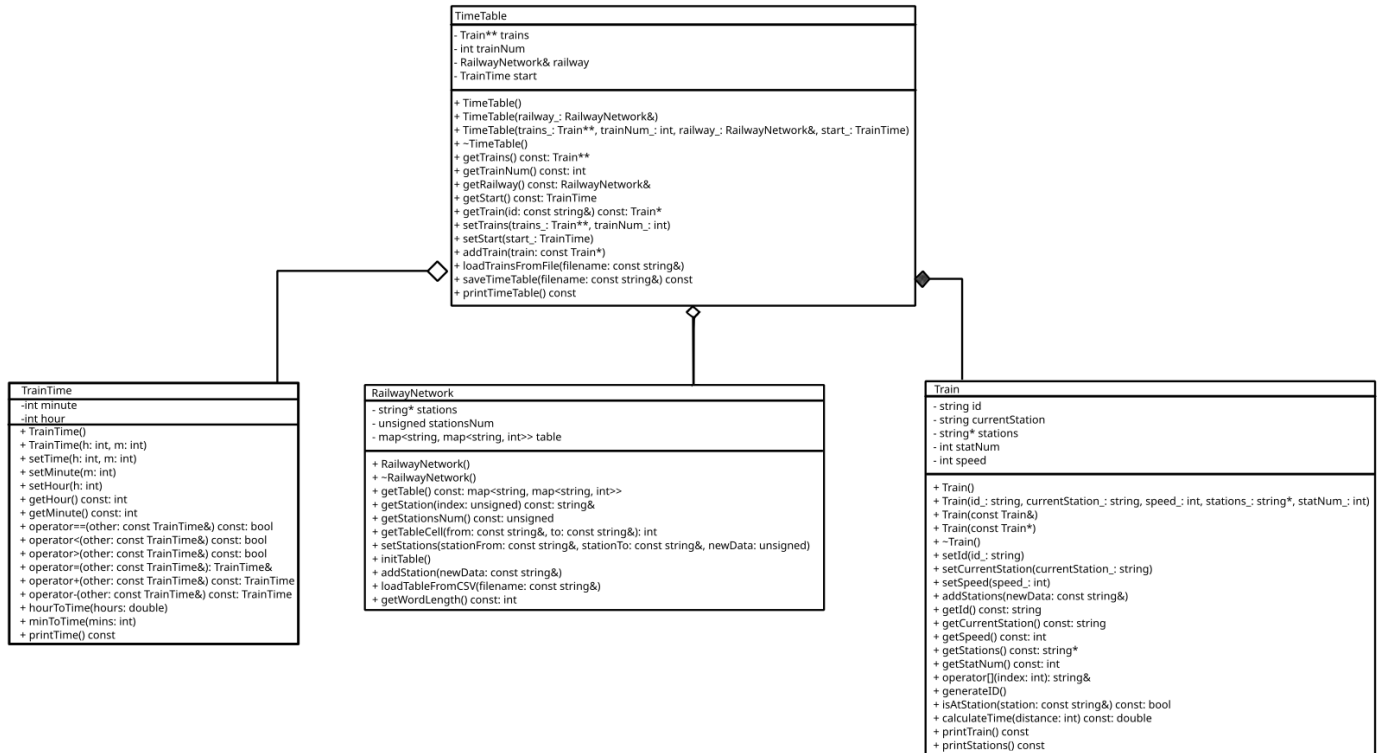
- int minute -> percek
- int hour -> órák

## Metódusok

### Publikus

- TrainTime() -> default konstruktor
- TrainTime(const int h, const int m) -> óra és perc konstruktor
- void setTime(const int h, const int m) -> beállítja az óra és perc tagokat
- void setMinute(const int m) -> beállítja a perc tagváltozót
- void setHour(const int h) -> beállítja az óra tagváltozót
- int getHour() const -> visszatér az órával
- int getMinute() const -> visszatér a perccel
- bool operator==(const TrainTime& other) const -> egyenlőséget dönt el két idő között
- bool operator<(const TrainTime& other) const -> relációt végez két idő között
- bool operator>(const TrainTime& other) const -> relációt végez két idő között
- TrainTime& operator=(const TrainTime& other) -> két időt tesz egyenlővé
- TrainTime operator+(const TrainTime& other) const -> két időt ad össze
- TrainTime operator-(const TrainTime& other) const -> két időt von ki egymásból
- void hourToTime(const double hours) -> egy órából egyszerűsít egész percekre
- void minToTime(const int mins) -> egy órából egyszerűsít egész percekre és órákra
- void printTime() const -> kiírja az időt vezető 0-al és 2 számjegyen :-al szeparálva az órát és percet

# UML osztálydiagramm





# Összegzés

## Mit sikerült és mit nem sikerült megvalósítani a specifikációból?

A specifikációt sikerült megvalósítani, egy menüt nem tudtam a határidő szűke miatt befejezni, az a startTime értékadás lett volna.


## Mit tanultál a megvalósítás során?

Nehézséget az idő jelentette, illetve a file beolvasó függvények elég komplexek lettek.

## Továbbfejlesztési lehetőségek

Több menüt lehetne hozzá adni, illetve a file kiírás többféle módo is lehetne.

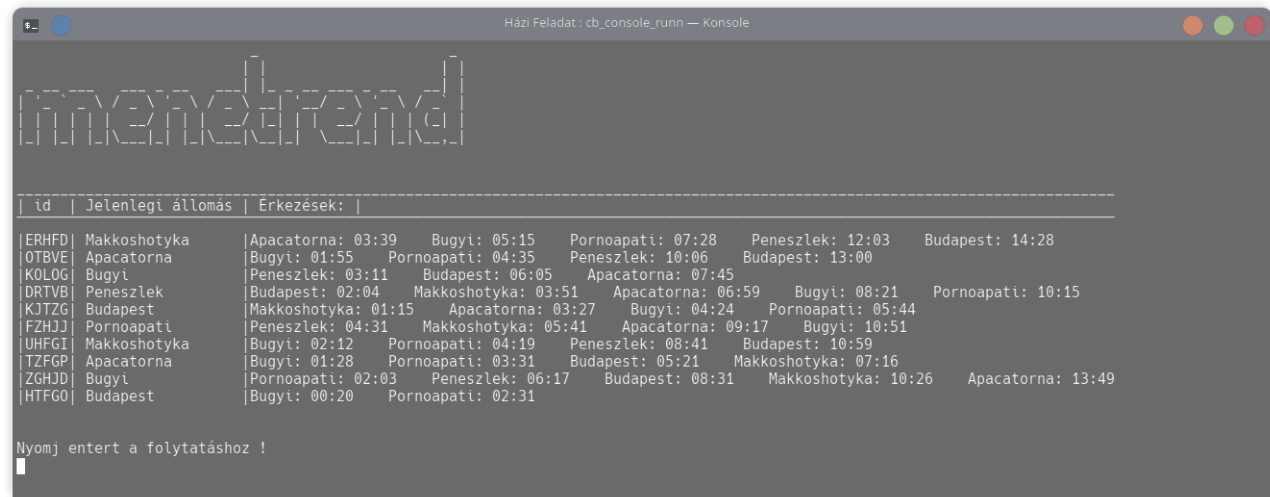
# Képernyőképek a futó alkalmazásról



```
Házi Feladat: cb_console_runn — Konsole

menetrend

Mit szeretnél csinálni?
1: kilratás a terminálba
2: kilratás fileba
3: vonat menetidő lekérdezés
0: kilépés
```



```
Házi Feladat: cb_console_runn — Konsole

menetrend

-----
| id | Jelenlegi állomás | Erkezések: |
-----
|ERHFD| Makkoshotyka      | Apacatorna: 03:39  Bugyi: 05:15  Pornoapati: 07:28  Peneszlek: 12:03  Budapest: 14:28
|OTBVE| Apacatorna        | Bugyi: 01:55  Pornoapati: 04:35  Peneszlek: 10:06  Budapest: 13:00
|KOLOG| Bugyi              | Peneszlek: 03:11  Budapest: 06:05  Apacatorna: 07:45
|DRTVB| Peneszlek          | Budapest: 02:04  Makkoshotyka: 03:51  Apacatorna: 06:59  Bugyi: 08:21  Pornoapati: 10:15
|KJTZG| Budapest           | Makkoshotyka: 01:15  Apacatorna: 03:27  Bugyi: 04:24  Pornoapati: 05:44
|FZHJJ| Pornoapati         | Peneszlek: 04:31  Makkoshotyka: 05:41  Apacatorna: 09:17  Bugyi: 10:51
|UHFGI| Makkoshotyka       | Bugyi: 02:12  Pornoapati: 04:19  Peneszlek: 08:41  Budapest: 10:59
|TZFGP| Apacatorna         | Bugyi: 01:28  Pornoapati: 03:31  Budapest: 05:21  Makkoshotyka: 07:16
|ZGHJD| Bugyi              | Pornoapati: 02:03  Peneszlek: 06:17  Budapest: 08:31  Makkoshotyka: 10:26  Apacatorna: 13:49
|HTFGO| Budapest           | Bugyi: 00:20  Pornoapati: 02:31

Nyomj entert a folytatáshoz !
```