Racetrack!

A versenyző csapat feladata egy olyan bot kifejlesztése, amely képes elvezetni egy autót egy szimulált versenypályán.

# Pálya

A pálya minden esetben egy téglalap, amely W x H négyzet alakú blokkra van felosztva.

Egy blokk lehet:

* Üres (a forráskódban 0 érték)
* Akadály (a forráskódban -1 érték)
* Ellenőrző pont (a forráskódban pozitív egész, 1…)

A blokk mérete pályáról pályára változhat.

Kezdéskor a bot a következő pályaadatokat kapja meg:

* A pálya oszlopainak száma ("racetrack\_columns *num*")
* A pálya sorainak száma ("racetrack\_rows *num*")
* Egy blokk oldalának a mérete pixelben ("block\_size *num*")
* Az akadályok helye a pályán ("block *x* *y*"),
* Az ellenőrző pontok helye a pályán ("checkpoint id *x* *y*") sorszámmal



Figure 1A játék; bal alsó sarokban "Player 1" a kezdőpozícióban

# Autó

Az autó egy téglalappal van modellezve, a főbb paraméterei:

* Hossza (*CarDynamics.h*)
* Szélessége (*CarDynamics.h*)
* Pozíciója (hol helyezkedik el a közepe: "car\_position *x* *y*")
* Sebessége (iteráció/tick alatt megtett hossz "car\_speed *speed*")
* Iránya (radiánban, "merre néz az eleje": "car\_direction *dir*")
* Kormányszög (radiánban, ~merre néz az első kerék, relatív az autó irányához képest: "car\_steering\_angle *angle*")

A virtuális autó minden esetben a pálya bal alsó sarkában lévő [0; 0]-ás blokk közepén kezd és felfelé néz (iránya PI/2).

A blokk-indexek innetől növekednek jobbra x dimenzióban, (W-1)-ig, felfelé y dimenzióban (H-1)-ig. Az autó koordinátája ennek megfelelően [blokk méret/2.0; blokk méret/2.0].

**Megjegyzés**: A pálya leírása blokk-indexekkel, míg az autó pozíciója dupla precizitású lebegőpontos (double) számpárokkal történik. Eképpen a pálya leírható egyfajta mátrixként, ami az autó által használt koordináta rendszerre a blokkmérettel skálázódik fel.

## Az Autó Dinamikája

Az autó mozgása a szimulációban egy kezdetleges dinamikai modellel van közelítve. A modell megtalálható implementálva a kiinduló bot forráskódjában forráskód kommentekkel ellátva.

**Megjegyzés**: A modell bármilyen formában használható, akár prediktív szabályozáshoz, akár valamilyen analitikus megoldás létrehozásához.

## Szimuláció

A szimuláció apró időközökre, *tick*-ekre végzi el a verseny állapotának kiszámítását. A botnak a feladata az egyes tick-ekben megvalósítandó akció meghatározása.

# A Verseny Szabályai

A pálya teljesítése az az összes ellenőrző pont sorrendben történő érintését jelenti.

A pálya teljesítése kudarcba fullad ha:

* Az autó érinti a pálya szélét
* Az autó nekimegy egy akadálynak
* Eltelik 5000 tick-nyi idő anélkül, hogy az autó sorrenben érintette volna az összes ellenőrző pontot

# A Bot

A versenyző botok rendelkezésére álló adatok minden tick-ben (iterációs lépésben):

* Az autó pozíciója, abszolút koordinátban, az origó a pálya bal alsó csúcsa
* Az autó orientációja, merre néz, radiánban
* Az autó sebessége
* Az autó kormányszöge (mennyire vannak elforgatva az első kerekek az autóhoz képest radiánban)

A bot által elvégezhető akciók:

* Sebesség befolyásolása (pedál)
  + Gáz adás
  + Fékezés
  + Egyik sem (enyhén lassul minimális súrlódás hatására)
* Kanyarodás (kormány)
  + Jobbra
  + Balra
  + A kormány elengedése (a kormányszög viszatér középre de fokozatosan csökkenve)

Az akciók hatásai az autóra a kezdő bot forráskódjában megtalálhatóak (a dinamikai modell implementációjában).

**Megjegyzés**: Egy botnak 100ms ideje van válaszolni a szimulátonak minden tick-ben, különben a szimulátor úgy veszi, hogy se nem ad gázt, se nem fékez és nem is kanyarodik aktívan. Az késve adott válasz elveszhet vagy a következő körben kerülhet feldolgozásra, ami további késéseket vagy további válaszok elvesztését jeneltheti.

**Figyelem**: Törekedni kell továbbá arra, hogy a bot teljesítménye a verseny során kiértékelhető legyen. Az olyan botok, amelyek gyakran vesznek igénybe sok időt eléggé lelassítják a szimulációt nem indulhatnak egymás ellen. A verseny feltétele, hogy a pályán as adott idő alatt (5000 tick) végig tudjanak menni.

## A használt programozási nyelv

**A botot elsősorban C nyelven megírva várjuk, de igény szerint lehetőség van Java-s bot használatára is.** A kezdő csomag mindenesetre csak C-s kezdőbot kódját tartalmazza.

A C-s implementáció esetén a sztandard C függvénykönyvtár (MinGW-s részhalmaza) használható.

Törekedni kell arra, hogy a beadott foráskód egyszerűen fordítható legyen.

# Pontozás

A versenyző botok pontot szerezhetnek pályák teljesítéséért, valamint egymás elleni verseny során elért helyezésért.

Egy pálya teljesítése azt jelenti, hogy a bot a rendelkezésre álló tick-ek alatt sikeresen végigmegy az összes ellenőrzőponton sorrendben, és nem ütközik falnak vagy a pálya szélének.

A versenyen a helyezések a pályát sikeresen teljesítő botok időeredménye alapján alakulnak ki.

**A helyszíni versenyen összesen 4 pálya lesz, melyek között szerepel a kezdő csomagban megtalálható pálya is.**

**Minden pálya teljesítése 20 pontot ér, részpontszám a kövekezőképpen kerül számításra:**

**Also egész rész (20 \* *<Teljesített ellenőrző pontok száma>* / *<Összes ellenőrző pont száma>*)**

Ha egy bot a kezdő csomag pályáján a 2. ellenőrző pontig jut, a csapat

alsó egész rész (20 \* 2 / 3) = 13 pontot fog kapni.

Az ismeretlen 3 pálya a verseny napján a helyszínen lesz publikálva, némi időt hagyva a csapatoknak a bot vezető program finomhangolására.

A versenyző csapatok botjai éles versenyekben is részt vesznek mind a 4 pályán, melyeken egyenként további plusz pontokat lehet szerezni:

1. helyezett: **+5 pont**
2. helyezett: **+3 pont**
3. helyezett: **+1 pont**

# Kezdő csomag

A kezdő csomag tartalmazza a hivatalos szimulátort, ami az éles versenyen is használva lesz.

A kezdő csomag tartalmaz egy kezdő pályát, amely felhasználható a bot teszteléséhez és a pontozásnál is szerepet kap. A pályát szimulátor olvassa be és az információkat róla ő osztja meg a botokkal, tehát a botoknak nem feladata a pálya beolvasása.

## A szimulátor használata

**A szimulátor minimum *Java 8*-as környezetet igényel** a futtatáshoz és a *racetrack.jar* fájl futtatásával indul.

A botot futtatható formában (1 fájl lehet) kell elhelyezni a szimulátor *bots* könyvtárában. A futtatható fájl neve: **<csapatnév>**[.<kiterjesztés>].

A szimulátor induláskor megnézi a bots könyvtárat és a benne található botokat elkezdi futtatni és kommunikál velük azok standard IO-ján keresztül.

**A szimulátor a szimulációt billenytű lenyomásra kezdi.**

## A kezdő bot

A kezdő csomag tartalmazza a C-s kezdő bot forráskódját, melybe beleértendő a kommunikáció a szimulátorral és az autó dinamikai modellje.

A C-s kezdő bot forráskódja a starterbot/c könyvtárban található.

**Megjegyzés:** van lehetőség *Java*-s bot készítésére is, de ez esetben a csapatok rendelkezésére nem áll kezdő bot. A kommunikációt a C-s kezdőbot mintájára maguknak kell elkészíteniük és futtatható *jar* formában elhelyezni a szimulátor *bots* könyvtárába.

## Futtatás emberi játékossal

A következő parancssori kapcsolóval létrehoz egy ‘Player 1’ nevű versenyzőt, amit billenyűzettel (kurzormozgató gombok) lehet irányítani:

--keyboardPlayer

## Futtatás más pályával

A következő parancssori kapcsolóval lehet másik pályát választani a *maps* könyvtárból:

--map=*<pálya fájl neve>*

# Pálya formátum

A pályát egy ‘sima’ szöveges fájl ír le egy vesszővel szeparált listaként. Az első két elem a pálya szélessége és magassága blokkokban, ezt követően soronként van felülről lefele megadva az egyes blokkok szerepe:

* 0 🡪 üres (aszfalt)
* X 🡪 akadály
* 1… 🡪 ellenőrző pont, ahol a szám az ellenőrzőpont *egyedi* sorszáma

# Debug

### Elvárt válaszidő beállítása

A következő parancssori kapcsolóval lehet átállítani azt a válaszidőt amit a szimulátor elvár az egyes botoktól:

--timeout\_ms=*<érték ms-ben>*

Ha az érték kisebb vagy egyenlő mint 0, a várakozás ideje nem lesz kolátozva, ‘végtelen’ ideig for a szimulátor várakozni a bot válaszára.

### Logolás

Jelenleg nincs szofisztikált log támogatás, a botoknak maguknak kell megoldania. Arra érdemes figyelni, hogy ilyesmi a végleges verzióban ne legyen (engedélyezve), mert jelentős overheaddel járhat.

# Lehetséges problémák és megoldásuk

## Nem megfelelő terminálásból fakadóan beragadt processzek

A taszk manager-ből kell őket terminálni.