I. 182.
Szabó Gábor (i1821.cpp)

A feladat megoldása során a következő megközelítést alkalmaztam: az utak egyes elemei ahol autók állhatnak, *mezők*. Ha foglalt egy mező, azaz áll rajta autó, akkor ez a mező "tudja" autójának célját és sebességét. A szimuláció egyes lépéseiben egyik mezőről a következőre pakolja az autókat a program.

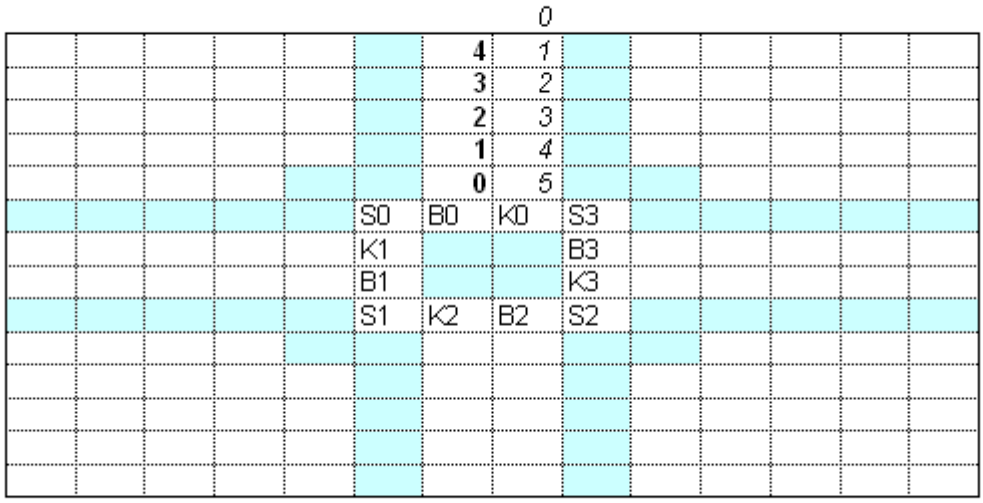
A pályán, ábrán vannak *kivezető utak, bevezető utak, körforgalomból kilépő, körforgalomba belépő, és sarokmezők*. Az ábrán a kivezető mezők vannak dőlttel szedve az 1-es út esetén, vastaggal a bevezető mezők, S jelöli a sarokmezőket, B a belépő, K pedig a kilépő mezőket.

A szimuláció során a program először a kivezető utakat vizsgálja meg, vajon van-e autó, amit el lehet tüntetni (ehhez van felvéve a 0. segédmező), majd megpróbálja a kivezető szakaszon előre léptetni a járműveket. Eztán jön a körforgalom vizsgálata. Először megpróbál kilépő mezőkről kiléptetni az útra, ha lehet. Majd sarokmezőről kilépőre, belépőről sarokra, majd kilépőről belépőre. Ehhez van felvéve egy logikai *kilephet* tömb, hogy egy szimulációs lépésen belül egy kilépőre rálépő autó ne mozduljon meg kétszer és kvázi "ugorja át" a kilépő mezőt.

Előfordulhat, hogy senki sem tud így a körforgalomban mozogni, ekkor a program "fordít" egyet a körforgalmon, vagyis mindenkit eggyel előrébb küld, feltéve, ha mindenki meg akar mozdulni abban a lépésben. Végül pedig a bevezető útszakaszt kell megvizsgálni, és ha kell egy új autót behozni.

Fejlesztőkörnyezet: Dev-C++ 4.9.9.2 (fut Borland C++ Builder 6 alatt is)

Megjegyzés: Szerintem a programhoz kellett a klasszikus gotoxy parancs, ez viszont nincs implementálva a Dev-C++ környezetben, vagyis ezt nekem kellett megoldani, ott van a forráskód elején. Szintén ott van a clrscr parancs, és a wait() függvény is, ami arra szolgált, hogy ne pillanatok alatt zajlódjanak az események.



1. ábra