

Imperatív Programozás Labor ZH – 2024.12.23.du.

Elvárások a programmal szemben

- A megoldáshoz semmilyen segédeszköz nem használható, kivéve a C referenciát.
- A program végleges verziójának működőképesnek kell lennie. Forduljon és fusson!
 - **A nem forduló kód 0 pontot ér!**
- Ne használj globális változókat! Csak a makrók megengedettek!
- Törekedj a szép, áttekinthető kódolásra, használj indentálást, kerülj a kódismétlést!
- Kommunikáljon a program! Legyen egyértelmű a felhasználó számára, hogy mit vár a program, illetve pontosan mi történik!
- Logikusan tagold a megoldást (használd függvényeket, külön fordítási egységeket)!
- Ne foglalj szükségtelenül memóriát, és kerülj a memóriaszivárgást!
- Kerülj a nem definiált viselkedést okozó utasításokat!
- Ahol nincs pontosan leírva egy feladatrész, annak egyedi megvalósítását rád bízunk.
- A struktúrák paraméterként átadásánál gondold át, hol érdemes mutatót használni!
- **A végleges programban a be nem tartott elvárások pontlevonással járnak!**

Busz megálló menedzser

A feladatod, hogy létrehozz egy programot, amely képes egy kisváros buszmegállóit számontartani, és egy leegyszerűsített térképen megjeleníteni, valamint útvonalakat megadni. A buszmegállónak tároljuk a nevét és a várostérkép koordinátáit, ahol elhelyezkednek. Tetszőlegesen új buszmegállót hozhatunk létre, vagy törölhetünk régiakat. Az egy városhoz tartozó buszmegállót .txt fájlban tároljuk és igény szerint betölthetjük.



A feladat összesen **50** pontot ér.

Legalább **10** pontot kell gyűjteni a tárgy sikeres teljesítéséhez.

A megoldásra **180 perc** áll rendelkezésre.

Beadáskor csak a .c, .h és .txt fájlokat kell becsomagolva feltölteni a következő formában: **<neptun_kód>.zip** (Az utoljára feltöltött megoldást pontozzuk.)

A következőkben felsorolásra kerülnek a kötelezően megvalósítandó alprogramok, de ezeken túl ajánlott további „segéd” alprogramok elkészítése is.

Főprogram – main() (5 pont)

Elsődleges feladata a program összefogása és kommunikáció a felhasználóval. Köszöntsük a felhasználót, és röviden ismertessük a programot. (Saját szavakkal, pár mondatban elegendő.) Ezután egy menü segítségével ajánljuk fel a lehetséges funkciókat a felhasználónak, nevezetesen:

- 1) Térkép – az aktuális város buszmegállójának megjelenítése térképes módban a sztenderd kimenten
- 2) Lista – az aktuális város buszmegállójának felsorolása szöveges módban
- 3) Új megálló – a felhasználó által új megálló létrehozása és hozzáadása az eddigiekhez
- 4) Megálló törlés – a felhasználó által megadott sorszámú buszmegálló törlése
- 5) Mentés – az aktuális város buszmegállójának fájlba történő másolása
- 6) Betöltés – a korábban fájlba mentett buszmegállók betöltése
- 7) Útvonal – a felhasználó által két kiválasztott állomás között útvonal rajzolása
- 8) Kilépés – a program befejezése

A program készüljön fel arra is, ha nem létező menüpontot adna meg a játékos.

1) Térkép - show_map() (5 pont)

Az aktuális várost mindig egy 10x10-es térképpel reprezentáljuk, melyet csak és kizárólag a megjelenítéshez fogunk használni. (A méretek tárolásához használjunk makrókat.)

Az alprogram készít egy leegyszerűsített térképet, melyen elhelyezi a 'B' karakterek által jelölt buszmegállókat a megfelelő pozíciókba. Minden egyéb térkép mező maradjon üres (space). A könnyebb áttekinthetőség kedvéért jelöljük kisbetűkkel a sorokat és számokkal az oszlopokat, hasonlóan a mellékelt ábrához.

Az alprogram jelezze a terminálon szövegesen, ha nincs megjeleníthető buszmegálló, és a program térjen vissza a menübe.

	1	2	3	4	5	6	7	8	9	10
a										
b		B								
c										B
d								B		
e										
f			B							
g					B					
h										
i								B		
j										

2) Lista - print_bus_stop() (2 pont)

Az alprogram egyszerűen kiírja az aktuális buszmegállókat felsorolásszerűen. (Sorszámmal ellátva, egy sorba egy megálló.)

Pl.: 1. Main Street (4, 8)
 2. Park Avenue (6, 3)
 ...

3) Új megálló - create_bus_stop() (7 pont)

Az alprogram lehetőséget biztosít új buszmegálló létrehozására. Először is kérje be a felhasználótól a megálló nevét és koordinátáit, melyek az 1-10-es skálán vehetnek fel értékeket. Hibás koordináta érték esetén figyelmeztessük a felhasználót és ismételjük meg az adat beolvasását.

Figyelj oda, hogy a megálló nevének végleges tárolására csak a szükséges memóriát foglald le!

Tipp: Ne felejtkezz el a buszmegállók tárolójának memória szervezési feladataival is foglalkozni!

4) Megálló törlés - delete_bus_stop() (6 pont)

Az alprogram felsorolja az aktuális buszmegállók listáját a terminálon, majd bekéri a törlendő sorszámát a felhasználótól. A megadott buszmegállót eltávolítja a listából. Fontos, hogy a lista implementációjának megfelelően kezeld le a keletkező „lyukat”.

Hibás index esetén visszatér a menübe.

Tipp: Érdemes a buszmegálló kiválasztást külön alprogramba szervezni.

Tipp 2: Figyelj oda a helyes memóriakezelésre!

5) Mentés - save_list() (4 pont)

Az aktuális buszmegálló lista mentése. Az alprogram bekéri a fájl nevét a felhasználótól, majd kimentti a buszmegállók listáját a fájlba. Egy buszmegálló két sorba fog kerülni. Az első sorba a megálló neve. A második sorba a koordinátái. Az alprogram jelezze a terminálon szövegesen, ha nincs menthető lista és a program térjen vissza a menübe.

Megjegyzés: A fájlnevek kezelésére használhatsz klasszikus tömböt.

6) Betöltés - load_list() (10 pont)

A felhasználó által megadott fájlból betölti a buszmegállók listáját. Fontos! Előfordulhatnak hibás koordinátájú buszmegállók (azaz nem az 1-10 tartományba esik az értékük), ezeket hagyjuk ki. A korrekt buszmegállókat töltjük csak be a fájlból.

Tipp: Figyelj oda a helyes memóriakezelésre!

Megjegyzés: A fájlnevek kezelésére használhatsz klasszikus tömböt, de a megállók neveinek tárolásához pontos memórafoglalás kell!

7) Útvonal - fastest_road() (11 pont)

A program kirajzol a felhasználó által kiválasztott két állomás között egy útvonalat. Ehhez érdemes a korábbi 'show_map()' függvényt tovább fejleszteni két további paraméterrel. Miután az útvonal alprogram bekérte a két érintett buszmegálló lista indexét (hasonlóan, mint a törlésnél történt) meghívja a térkép kirajzolót az indexekhez tartozó buszmegállók pontereivel kiegészítve.

Megjegyzés: a show_map() korábbi funkciója maradjon továbbra is működőképes, csak abban az esetben NULL pontereket adjunk neki.

Az útvonal kirajzolást ágyazzuk bele a térkép megjelenítésbe. Maga az útvonal úgy nézzen ki, hogy egy vízszintes ('-' elemek), egy függőleges ('|' elemek) részből, illetve egy 'x' sarok pontból álljon, ami összeköti a két megadott megállót. Ügyeljünk arra, hogy az esetleg útba eső megállót ('B') ne írjuk felül.

A mellékelt képen az 1. és 2. ábra egy-egy lehetséges megoldást mutat ugyan arra az esetre. A 3. ábra egy függőleges útvonal részt mellőző példát mutat.

	1	2	3	4	5	6	7	8	9	10
a										
b		B	-	-	-	-	-	X		
c										B
d								B		
e										
f			B							
g					B					
h										
i							B			
j										

	1	2	3	4	5	6	7	8	9	10
a										
b		B								
c									B	
d	X	-	-	-	-	-	B			
e										
f			B							
g					B					
h										
i							B			
j										

	1	2	3	4	5	6	7	8	9	10
a										
b		B								
c										B
d									B	
e										
f			B	-	X					
g					B					
h										
i							B			
j										

Tipp: az útvonal kirajzolásának megtervezésekor használjuk az azonos koordináták különbségét, de számítsuk bele az előjelet is!