Klases ekskursiju transporta plānošanas programma

# Programmas vispārīgs apraksts

* Programmas mērķis – aprēķināt ekskursijas **kopējās izmaksas un laiku**, lai nokļūtu starp **vairākām pieturām**.
* Pieturas, to attālumu, un laiku starp tām **lietotājs ievada pats**.
* Ekskursija, ko ievada lietotājs, sastāv no **vairākām pieturām, pilsētām vai vietām.**
* Papildus kopējām izmaksām un laikam, programma arī izvadīs **izmaksas uz vienu skolēnu**.
* **Pirms aprēķiniem**, programma **dod iespēju rediģēt datus**.
* Definētie transporta **veidi** – autobuss, vilciens, iešana ar kājām.

# Specifikācija

## Sākums

Programma sākumā izdrukā nosaukumu.

## Kādi ievaddati tiks doti?

1. Saraksts ar pieturām (jeb vietām, kur klase dosies)
   * Tips – *list*, kas sastāv secībā *[pietura, transports, pietura, …]*.
   * *transports* ir *list*, kurš satur informāciju par pieturām abās pusēs tam:
     + 1. Posma attālumu, tips - *float*
       2. Transporta veidu, tips – *str*
       3. Transporta laiku, tips - *float*
       4. Biļetes cena*,* tips - *float*
   * Saglabāts mainīgajā ar nosaukumu *posmi*
2. Skolēnu skaits
   * Tips – *int*
   * Saglabāts mainīgajā ar nosaukumu *n\_skoleni*

### Kā tiks iesniegti ievaddati?

* Visi ievaddati tiek saņemti ar funkciju *pareiza\_ievade()*.
* Nepareizas ievades gadījumā, programma prasa lietotājam datus ievadīt vēlreiz, kamēr tie tiek ievadīti pareizi.
* Pieturu/pilsētu/vietu nosaukumi tiek **saglabāti ar mazajiem burtiem, bet izdrukāti ar lielo pirmo burtu.**

Lai iegūtu pieturas, programma lūdz lietotājam **vienā rindiņā** ievadīt **ar komatiem atdalītus** pieturu nosaukumus.

Pēc tam, programma **par katru posmu** pieprasa **attālumu, transporta veidu, un biļetes cenu. Ja transporta veids ir iešana ar kājām,** tad cenu neprasa, un iestata uz 0. **Transporta viedu var pieprasīt saīsināti** – nevis “vilciens”, bet “v”, bet sarakstā tam jābūt pilnā garumā (skat. Atbilžu varianti)

#### Kādi ir iespējamie atbilžu varianti un mērvienības?

* Posma attālums – km, kilometri
* Transporta veids – viens no [“vilciens”, “autobuss”, “kājām”]
* Biļetes cena – eur, eiro, €
* Transporta laiks – h, stundas

Mērvienību gadījumā tās tiek izdrukātas blakus izvaddatiem.

Kad visi ievaddati saņemti, programma prasa, **vai lietotājs vēlas veikt izmaiņas.**

**Ja lietotājs vēlas veikt izmaiņas:**

Rediģēšanu veic ar funkciju *rediget\_posmu().*

## Kā aprēķināt rezultātu?

Cenu vienam skolēnam programma aprēķina **summējot visas individuālās biļetes cenas katram posmam, un saglabā mainīgajā** *cena\_skolenam* *(float)*.

Cenu **visiem skolēniem** *programma iegūst sareizinot cena\_skolenam* ar *n\_skoleni*.

## Kādas funkcijas?

Programmā jābūt šādām funkcijām:

### *pareiza\_ievade(derigas\_atbildes: list)*

* Izsauc *input()* funkciju, un pēc tam pārbauda, vai lietotāja sniegtā atbilde ir atrodama parametrā *derigas\_atbildes*.
  + Ja nav atrodama, funkcija izdrukā paziņojumu, ka atbilde nav derīga, un prasa vēlreiz.
  + Ja ir atrodama, funkcija neko neizdrukā un ar *return* atgriež ievadīto vērtību.
* Ievades teksts lietotājam parāda iespējamos atbilžu variantus, vai arī **mērvienību, kuru programma sagaida.** (skat. Atbilžu varianti)

### *posms(sakums, beigas)*

* Funkcija lietotājam prasa visu, kas minēts pirmajā punktā, pie “Kādi ievaddati tiks doti?”
* Šos datus funkcija ievieto sarakstā, tādā secībā, kā aprakstīts pirmajā punktā.
* Pēc tam, funkcija šo sarakstu ievieto starp pieturām, sarakstā *posmi.*

### *rediget()*

* Funkcija lietotājam dod izvēli – vai viņš vēlas rediģēt pieturu, vai transportu.
  + Ja lietotājs izvēlas rediģēt pieturu, viņam piedāvā visas pieturas ar to attiecīgo indeksu. Lietotājs izvēlas indeksu, kuram vēlas mainīt nosaukumu, un programma atjaunina sarakstu.
  + Ja lietotājs izvēlas rediģēt transportu, viņam prasa divas pieturas, starp kurām transports tiks rediģēts.
    - Pēc tam, tiek prasīta jauna informācija katrai transporta īpašībai (attālums, laiks, cena, u.tml).
    - Ja lietotājs sniedz tukšu ievadi, dati tiek paturēti tādi paši.
    - Ja posms starp divām pieturām nepastāv, programma izdrukā kļūdu un prasa lietotājam vēlreiz izvēlēties posmu.

## Kādi izvaddati tiks doti?

**Visi ievades kļūdu paziņojumi ir skaidri izcelti**, tā, lai tie atšķirtos no citiem izvades datiem, piemēram – ar izsakuma zīmi rindiņas sākumā.

**Pieturu nosaukumi tiek pārviedoti uz lielajiem sākumburtiem**, un visas float vērtības **noapaļotas līdz diviem cipariem aiz komata**.

Ja lietotājs nevēlas veikt izmaiņas, un visi dati ir ievadīti pareizi, programma **uz ekrāna**izvada ar attiecīgajām mērvienībām (skat. Atbilžu varianti):

* Sākumpunktu un galamērķi, kopējo laiku, kopējās izmaksas, kopējo attālumu un **izmaksas uz vienu skolēnu**.
* Informāciju par katru posmu:
  + Attālums
  + Laiks
  + Transporta veids
  + Cena

## Beigas

Pēc gala atbildes izvadīšanas, **programma paziņo par beigšanu un iziet**. Dati nekur **netiek saglabāti**.

## Piemērs

[ Klases ekskursiju transporta plānošanas programma ]

Lūdzu, ievadiet ar komatiem atdalītu pieturu sarakstu [piem. Rīga,Sigulda]: Rīga,Liepāja

Lūdzu, ievadiet skolēnu skaitu [>0]: -3

! Nepareiza atbilde! Lūdzu, mēģiniet vēlreiz.

Lūdzu, ievadiet skolēnu skaitu [>0]: 16

[ Posmi ]

( Posms Rīga-Liepāja )

Ievadiet attālumu [km]: 150.5

Ievadiet laiku [h]: 3.2

Ievadiet transporta veidu [v – vilciens, a – autobuss, k – kājām]: v

Ievadiet transporta cenu [€]: 11.42

( Visi posmi ievadīti )

Vai vēlaties veikt izmaiņas [j – jā, n – nē]: n

[ Ekskursija ]

Sākums: Rīga | Beigas: Liepāja | Kopējais laiks: 3.2 h

Cena vienam skolēnam: 11.42€ | Cena 16 skolēniem: 182.72€

@ Rīga

|

[ Vilciens, 3.2 h, 11.42€

|

@ Liepāja

[ Paldies! Programma beidzas. ]