Tema 1

Data postării: 28.11.2024
Deadline: 20.12.2024 ora 23:59

1 Specificații

Se dorește implementarea unui sistem care să funcționeze conform următoarelor specificații:

- Orașul este împărțit în `nrC` cartiere.
  - Un **cartier** este o structură ce conține:
    - `id` – un întreg;
    - `nume` – un șir de caractere (`char*`) ce nu poate conține spațiu.
- Un **pachet** este reprezentat printr-o structură ce conține următoarele câmpuri:
  - `id` – un întreg;
  - `adresă` – un vector de 18 elemente (`int[18]`) conținând doar valori 0 sau 1;
  - `idCartier` – un întreg;
  - `strada` – un întreg;
  - `număr` – un întreg;
  - `prioritate` – o valoare din mulțimea {1, 2, 3, 4, 5}, unde:
    - 5 = prioritate maximă;
    - 1 = prioritate minimă.
  - `greutate` – un număr real (`float`);
  - `mesaj` – un șir de caractere (`char*`) reprezentând un scurt mesaj care însoțește pachetul:
    - Mesajul poate conține doar litere, cifre, spații și semne de punctuație (`.,!?:`) și se termină cu un newline (`'\n'`).
  - `codificare mesaj` – un întreg calculat după distribuirea pachetului (task I).
- Fiecare pachet are o adresă reprezentată printr-un vector de 18 poziții ce reține doar valori 0 sau 1, cu următoarea semnificație:
  - Primele 5 poziții reprezintă id-ul cartierului, următoarele 5 reprezintă strada, iar următoarele 8 reprezintă numărul.

Observație

Pentru câmpurile de tip `char*` (`nume_cartier` şi `mesaj`) **trebuie să alocați spațiu exact cât este necesar, nu mai mult**!

- Fiecare poștaș va avea un id ce coincide cu id-ul cartierului de care este responsabil, el trebuind să distribuie pachetele din acel cartier. Structura prin care e definit poștașul conține următoarele câmpuri:
  - `id` – un întreg din intervalul [0, 31], unde 32 este numărul maxim de poștași;
  - `nrPachete` – un întreg din intervalul [0, 49], unde 50 este numărul maxim de pachete distribuite de un poștaș;
  - un vector cu pachetele pe care le are de distribuit.
- Există un **poștaș șef** care primește inițial toate pachetele și are responsabilitatea de a le distribui către poștașii simpli, astfel încât fiecare să primească pachetele corespunzătoare cartierului său.
- După primirea pachetelor, poștașii:
  1. **Ordonează pachetele** de care sunt responsabili în funcție de:
     - **Prioritate** – pachetele cu prioritate mai mare sunt distribuite primele;
     - **Greutate** – dacă două pachete au aceeași prioritate, pachetul cu greutatea mai mare este distribuit primul.
  2. **Codifică mesajul** primit împreună cu pachetul după următorul algoritm:
     - Inversează ordinea cuvintelor din propoziție, eliminând toate semnele de punctuație.
     - Exemplu: dacă mesajul inițial este `"Ana, are cel mult 3 mere."`, după prelucrare, mesajul devine `"mere3multcelareAna"`.
  3. **Calculează un cod** care este suma produselor dintre codul ASCII al fiecărui caracter și poziția pe care acesta se află în interiorul șirului, după ce s-au efectuat prelucrările anterioare.
     - Numerotarea pozițiilor caracterelor începe de la 0.
  4. **Codul final** este restul împărțirii codului anterior la produsul dintre numărul casei și numărul străzii pachetului respectiv, la care se adună 1.

$$cod \ \textbf{MOD} \ (casa * strada + 1)$$

Poștașul șef va verifica dacă fiecare cod primit este corect, deoarece unii poștași obișnuiesc să modifice mesajele din anumite pachete. Verificarea codurilor o face folosindu-se de același algoritm pe care l-au folosit și destinatarii pentru a genera codul pentru fiecare pachet.

2 Cerința I

1. (20 p) Să se implementeze o funcție care citește de la tastatură următoarele date:
   - `nrC` – numărul de cartiere și de poștași;
   - Pe următoarele `nrC` linii se primește numele fiecărui cartier;
   - `nrP` – numărul total de pachete;
   - Pe următoarele `4 * nrP` linii se va primi fiecare pachet (câte patru linii / pachet):
     - adresa pachetului (18 valori 0 sau 1);
     - prioritatea;
     - greutatea;

– mesajul – acesta are lungimea de maxim 100 de caractere.

Atât `id`-urile cartierelor cât și cele ale pachetelor se vor completa automat începând de la valoarea 0. Nu se vor citi de la tastatură!

2. (15 p) Să se implementeze o funcție care, pentru un pachet dat, să extragă din adresa acestuia:

   - cartierul;
   - strada;
   - numărul;

   completând câmpurile corespunzătoare din structura pachetului.

3. (15 p) Să se implementeze o funcție care distribuie pachetele poștașilor.

4. (20 p) Scrieți o funcție care, primind un vector de structuri de tip pachet, să ordoneze pachetele din vector:

   - în funcție de prioritate (descrescător);
   - dacă mai multe pachete au aceeași prioritate, să fie ordonate după greutate (tot descrescător);
   - dacă există pachete cu aceeași greutate și aceeași prioritate, acestea să fie lăsate în ordinea naturală.

5. (20 p) Scrieți funcțiile pentru a calcula codificarea mesajului:

   (a) funcție care să inverseze ordinea cuvintelor dintr-un text și să elimine semnele de punctuație;

   (b) funcție care primește ca parametru un pachet și calculează codul mesajului conform algoritmului specificat mai sus.

3 Cerință II

6. (30 p) Scrieți o funcție care, primind `id`-ul poștașului, să altereze doar acele coduri care au cel puțin o cifră care se regăsește în `id`-ul poștașului.

   - Exemplu: pentru poștașul cu `id`-ul 6, acesta va altera codul 246, dar nu va altera codul 281 (fiindcă primul conține cifra 6, iar al doilea, nu).
   - Dacă poștașul are `id`-ul 13, atunci el va altera codul 4913, dar nu va altera codul 408.

   Se va folosi o funcție auxiliară care să altereze (modifice) codul unui mesaj după următorul algoritm:

   (a) Se calculează factorii primi ai `id`-ului poștașului. Acești factori primi vor reprezenta pozițiile biților modificați.
   - Dacă `id`-ul este 0, atunci vom considera doar valoarea 0 ca "*divizor*".
   - În cazul lui 1, singurul divizor va fi considerat 1.

   (b) Fiecare bit aflat pe pozițiile descrise la pasul anterior va fi "inversat" (1 devine 0, 0 devine 1).

- Biții se vor modifica o singură dată, chiar dacă factorul apare la o putere mai mare decât 1.
- Dacă există factori mai mari decât 31, atunci acei factori se vor ignora pe parcursul algoritmului.

(c) Numerotarea biților începe de la 0 și se face de la dreapta la stânga.

Funcția primește ca parametri codul și `id`-ul unui poștaș.

7. (10 p) Scrieți o funcție pentru atribuirea unui scor fiecărui poștaș, scorul fiind egal cu numărul de pachete distribuite corect / numărul total de pachete distribuite.

   - Un pachet se consideră distribuit corect dacă este corect codul corespunzător lui.

Observație

Considerăm că dacă un poștaș nu are niciun pachet de livrat acesta va avea scorul 0.

4 Formatul fișierelor

4.1 Input

Formatul unui fișier de intrare este următorul:

1. numărul cerinței care urmează să fie rezolvată (1-7)
2. `nC` = numărul de cartiere / poștași
3. `nC` linii cu numele cartierelor (pentru fiecare nume avem un șir de caractere fără spații)
4. `nP` = numărul de pachete
5. cate 4 linii pentru fiecare pachet:
   (a) pe prima linie: 18 numere întregi, 0 sau 1, pe o singură linie, despărțite prin câte un spațiu;
   (b) pe a doua linie: prioritatea (un număr natural între 1 și 5)
   (c) pe a treia linie: un număr real ce reprezintă greutatea (`float`);
   (d) pe a patra linie: un șir de caractere ce reprezintă mesajul.

4.2 Output

1. **Cerința 1**
   - se afișează pe câte o linie separată toate cartierele în format `id_cartier nume_cartier`;
   - se afișează toate pachetele, 4 rânduri pentru fiecare pachet:
     (a) `idPachet`
     (b) adresa (spațiu între caracterele din adresă)
     (c) prioritate greutate (cu 3 zecimale)
     (d) mesaj

2. **Cerința 2**
    - se afișează adresa fiecărui pachet pe 2 linii:
        (a) `idPachet`
        (b) `idCartier strada numar`
3. **Cerința 3**
    - se afișează câte 2 linii pentru fiecare poștaș:
        (a) `idPostas nrPachete`
        (b) `idPachet1 idPachet2 ... idPachetnr`
4. **Cerința 4**
    - afișarea ca la 3, doar că pachetele sunt acum sortate
5. **Cerința 5**
    - se afișează, pentru fiecare poștaș:
        – `idPostas nrPachete`
        – nrPachete linii care conțin: `idPachet codMesaj`
6. **Cerința 6**
    - afișare ca la 5
7. **Cerința 7**
    - câte o linie pentru fiecare poștaș de tipul: `idPostas scor` (3 zecimale)

5 Exemplu

5.1 Input

```
1
2
Cartier1
Cartier2
3
0 0 0 0 0 1 1 1 0 0 0 1 1 0 0 0 1 0
5
10.3
Cel din urma pachet
0 0 0 0 0 1 0 1 0 1 0 1 0 0 0 0 1 0
5
12.312
Al treilea pachet
0 0 0 0 1 1 0 1 0 1 1 1 1 0 0 0 1 0
2
22.3
Primul, pachet!
```

5.2 Output

1. **Cerința 1**

```
0 Cartier1
1 Cartier2
0
0 0 0 0 0 1 1 1 0 0 0 1 1 0 0 0 1 0
5 10.300
Cel din urma pachet
1
0 0 0 0 0 1 0 1 0 1 0 1 0 0 0 0 1 0
5 12.312
Al treilea pachet
2
0 0 0 0 1 1 0 1 0 1 1 1 1 0 0 0 1 0
2 22.300
Primul, pachet!
```

2. **Cerința 2**

```
0
0 28 98
1
0 21 66
2
1 21 226
```

Explicații:

- `idCartier` (codificat în primii 5 biți din adresă).
- `strada` (codificată în următorii 5 biți, adică pozițiile 5-9).
- `număr` (codificat în ultimii 8 biți, pozițiile 10-17).

(a) pachetul 0: [0 0 0 0 0] [1 1 1 0 0] [0 1 1 0 0 0 1 0]
idCartier = 0; strada = $2^2 + 2^3 + 2^4 = 28$; număr = $2^1 + 2^5 + 2^6 = 98$

(b) pachetul 1: [0 0 0 0 0] [1 0 1 0 1] [0 1 0 0 0 0 1 0]
idCartier = 0; strada = $2^0 + 2^2 + 2^4 = 21$; număr = $2^1 + 2^6 = 66$

(c) pachetul 2: [0 0 0 0 1] [1 0 1 0 1] [1 1 1 0 0 0 1 0]
idCartier = $2^0 = 1$; strada = $2^0 + 2^2 + 2^4 = 21$; număr = $2^1 + 2^5 + 2^6 + 2^7 = 226$

3. **Cerința 3**

```
0 2
0 1
1 1
2
```

Explicații:

- poștașul 0 trebuie să livreze 2 pachete, pachetele cu id 0 și 1;
- poștașul 1 trebuie să livreze 1 pachet, pachetul cu id 2.

4. **Cerința 4**

```
0 2
1 0
1 1
2
```

Explicații:

- poștașul 0 trebuie să livreze 2 pachete;
- conform ordonării, va trebui să livreze mai întâi pachetul 1 și apoi pachetul 0.

5. **Cerința 5**

```
0 2
1 841
0 1236
1 1
2 2301
```

Explicații:

- poștașul 0 va livra 2 pachete:
  - pachetul 1, codul mesajului 841
  - pachetul 0, codul mesajului 1236
- poștașul 1 va livra un pachet:
  - pachetul 2, codul mesajului 2301

6. **Cerința 6**

```
0 2
1 841
0 1236
1 1
2 2303
```

Explicații:

- Inițial, codul pachetului cu ID 2 era 2301.
- Fiindcă 2301 conține una dintre cifrele care se regăsesc și în ID-ul poștașului (1), codul a trebuit să fie alterat/modificat:
  - Factorii primi ai ID-ului poștașului: 1
  - 2301 în binar: 100011111101

- Penultimul bit al lui 2301 (poziția 1, de la dreapta la stânga) este inversat
- 1000111111[**0**]1 → 1000111111[**1**]1 = 2303
- 2301 → 2303

7. **Cerința 7**

```
0 1.000
1 0.000
```

- Poștașul 0 a reușit să livreze corect toate pachetele:
  - Pachetul 0 → corect
  - Pachetul 1 → corect
- Poștașul 1 nu a reușit să livreze corect niciun pachet:
  - Pachetul 2 → alterat

6 Precizări

- Separați logica programelor în mai multe funcții.
- **NU** se vor puncta sursele în care tot programul este scris în `main`!
- Este interzisă folosirea **variabilelor globale**! Folosirea acestora va fi depunctată.
- Soluția temei va fi scrisă în ANSII C! Nu folosiți sintaxă sau instrucțiuni specifice limbajului C++.
- Va trebuie să **creați un fișier README** în care să precizați numele, grupa, cât timp v-a luat implementarea cerințelor și explicați, pe scurt în câteva fraze, implementarea temei (comentariile din cod vor documenta mai amănunțit rezolvarea).
- Este recomandat ca liniile de cod și cele din fișierul README să nu depășească 80 de caractere.
- Temele sunt strict individuale. Copierea temelor va fi sancționată cu punctaj 0 pentru toți cei care au porțiuni de cod identice!
- Persoanele cu porțiuni de cod identice **NU** vor primi niciun punctaj pe temă.
- **NU** copiați cod de pe Internet! Se poate ajunge la situația în care doi studenți să aibă același cod, preluat de pe Internet, caz în care ambele teme vor fi punctate in TOTALITATE cu 0, deși studenții nu au colaborat direct între ei.
- **NU** folosiți tool-uri care generează automat cod pentru rezolvarea temei. Temele vor fi comparate cu soluțiile generate de astfel de tool-uri.
- Temele trimise după deadline **NU** vor fi luate în considerare.
- Datele de intrare sunt descrise pentru fiecare cerință individual și vor fi citite de la tastatură.

- Toate citirile se fac de la tastatură în codul final al temei! Folosiți fișiere sau/și redirectarea intrării/ieșirii doar pentru testele voastre intermediare!

  Operatorul `<` poate fi folosit pentru a redirecționa conținutul unui fișier către intrarea standard a unui program. Spre exemplu, considerăm că avem executabilul `pachete` (care în mod normal citește de la tastatură) și un fișier numit `input.txt`. Putem utiliza următoarea comandă pentru a redirecționa conținutul fișierului `input.txt` către intrarea executabilului `pachete`:

  ```
  ./pachete < input.txt
  ```

  De asemenea, putem redirecționa intrarea unui program și să salvăm ieșirea într-un fișier folosind operatorii `<` și `>` în același timp:

  ```
  ./pachete < input.txt > output.txt
  ```

7 Coding Style

Folosiți un coding style astfel încât codul să fie ușor de citit și înțeles. De exemplu:

- Dați nume corespunzătoare variabilelor și funcțiilor.
- Nu adăugați prea multe linii libere sau alte spații goale unde nu este necesar:
  - nu terminați liniile în spații libere, trailing whitespaces;
  - nu adăugați prea multe linii libere între instrucțiuni sau la sfârșitul fișierului.
- Principalul scop al spațiilor este identarea.
- Fiți consecvenți în coding style-ul ales.
- Vă recomandăm să parcurgeți această resursă: Coding style
- Există programe sau extensii pentru editoare text care vă pot formata codul. Este permisă utilizarea lor.
- Deși vă pot ajuta destul de mult, ar fi ideal să încercați să respectați coding style-ul pe măsură ce scrieți codul.

8 Trimiterea temei

Veți trimite o arhivă **ZIP** cu numele **EXACT** acesta: `GRUPA_Nume_Prenume_Tema1.zip`
De exemplu: `311CC_Popescu_Maria_Tema1.zip`
Arhiva va conține următoarele fișiere:

1. README
2. Makefile
3. pachete.c

Atenție!

Veți fi depunctați complet pentru formatarea incorectă a arhivei (alt nume, alt tip, alte nume pentru fișiere, alte fișiere în plus sau în minus etc.) - **0 puncte pe temă**.

- Pentru întrebări legate de temă se va folosi în mod exclusiv **FORUM-ul** temei, pe care vă recomandăm să îl vizitați chiar și dacă nu aveți întrebări, întrucât este posibil să aflați informații noi din întrebările puse de colegii voștri, respectiv din răspunsurile date de noi.
- Compilarea nu ar trebui să producă warning-uri (verificați prin adăugarea flagului -Wall la gcc).

Atenție!

- Temele trebuie să fie încărcate pe **vmchecker**. NU se acceptă teme trimise pe e-mail sau altfel decât prin intermediul **vmchecker-ului**.
- Link vmchecker: https://vmchecker.cs.pub.ro/ui/#PC-CC
- De asemenea, este disponibil un slot de încărcare și pe Moodle, ca soluție de backup. **Rezultatul final al temei va fi cel de pe vmchecker. Temele care nu sunt încărcate pe vmchecker nu vor fi luate în considerare.**

9 Punctaj

O temă perfectă valorează **130 de puncte**.
Punctajul pe cerințe este următorul:

Cerința	Punctaj
Cerința 1	90 puncte
Cerința 2	40 puncte

Atenție!

Depunctările care se pot aplica:

- **-10 puncte** pentru lipsa explicațiilor din README
- **-10 puncte** pentru lipsă comentarii din cod (atenție! nu trebuie comentată fiecare linie, doar ceea ce este esențial pentru a putea fi ușor de înțeles rezolvarea – detalii la curs!)
- **-20 puncte** pentru Coding Style necorespunzător
- **Depunctare totală pentru formatarea incorectă a arhivei** (alt nume, alt tip, alte nume pentru fișiere, alte fișiere în plus sau în minus) - **0 puncte pe temă**

O temă care NU compilează va fi punctată cu 0.
Vor exista mai multe teste pentru fiecare problemă în parte. Punctele pe teste sunt independente, punctajul pe un anumit test nefiind condiționat de alte teste.
În fișierul README va trebui să descrieți pe scurt soluția pe care ați ales-o pentru fiecare problemă și alte lucruri pe care le considerați utile de menționat

10 Checker

- Arhiva se va trimite pe vmchecker, unde tema se va testa automat.
- Pentru testarea locală, aveți disponibil un set de teste și un checker local.
- Punctajul acordat pe rularea testelor este cel de pe vmchecker.
- **Corectorii își rezervă dreptul de a scădea puncte pentru orice problemă găsită în implementare, dacă vor considera acest lucru necesar.**

Atenție!

Numele fișierului sursă trebuie să fie **pachete.c** pentru a putea fi verificat de checker. De asemenea, executabilul va avea numele **pachete**. Recomandăm folosirea fișierului Makefile din arhiva de checker.

Observație

Pentru a rula checker-ul trebuie să oferiți permisiuni fișierului **check.sh** (de exemplu, chmod a+x check.sh). Testarea implementării se va face utilizând **./check.sh**. De asemenea, tema va fi testată automat și la încărcarea pe **vmchecker**, așa cum este menționat în secțiunile următoare.