SO 2020 – Seria 1

Tema 1: Gestiunea fișierelor și a directoarelor

1. Descrierea temei

În cadrul acestei teme ne vom focaliza pe analiza **secvențelor de numere aleatoare** stocate în fișiere sub formă de succesiune de biți de 0 și 1. Astfel, orice fișier de intrare va primi programul nostru în vederea analizării, vom trata conținutul doar ca o secvență de biți.

O secvență de numere aleatoare este o aproximare a unui șir perfect aleator, care prezintă următoarele caracteristici importante:

- *Uniformitatea* valorilor: în secvențele șirului aleator toate valorile posibile apar cu aproximativ aceeași probabilitate
- *Impredictibilitate*: pe baza unei subsecvențe cunoscute, nu putem prezice următoarele valori din șir
- *Lipsa șabloanelor repetitive și a regularităților*
- Și multe altele – o lista posibil infinită de alte caracteristici.

2. Specificațiile temei

Scrieți un program C denumit **t1.c** care implementează următoarele cerințe minimale:

- Compilarea fără erori este o condiție esențială pentru acceptarea și evaluarea temei
  - Folosiți comanda: `gcc -Wall t1.c -o t1`
  - Warning-urile trebuie eliminate. Dacă totuși apar, vor duce la o penalizare de 10%
- Rularea se va face folosind următorul șablon: `./t1 [OPTIONS] [PARAMETERS]`
  - Opțiunile și parametrii sunt specificați la fiecare cerință
  - Rezultatul va conține pe prima linie SUCCESS sau ERROR și pe următoarele linii rezultatul efectiv, respectiv motivul eșecului.

1. Histograma pe N biți

- **(10p)** Determinați și afișați frecvența de apariție a tuturor valorilor posibile pe N biți în secvența analizată
  i. de exemplu, dacă N=1, atunci se afișează de câte ori apare bitul 1 și de câte ori apare bitul 0 în secvență.
  ii. Dacă N=2, atunci se afișează frecvența de apariție pentru fiecare dintre valorile: 00, 01, 10, 11.
- N poate lua valorile: 1, 2, 4 și 8.
- Formatul apelului:
  `./t1 histogram bits=<number_of_bits> path=<file_path>`
- Formatul rezultatului:
  i. Rezultat cu succes

Posibil rezultat pentru histograma pe 2 biți
SUCCESS 00: 3267 01: 3280 10: 3262 11: 3279

  ii. Rezultat cu eșec

Posibil rezultat pentru histograma pe 17 biți
ERROR Invalid number of bits Supported values are: 1, 2, 4, 8

Posibil rezultat pentru cale invalidă
ERROR Invalid file path

2. **Ce mai lungă subsecvență de 1-uri consecutive**
   - **(20p)** Determinați cea mai lungă subsecvență de biți de 1 consecutivi și afișați lungimea și poziția din cadrul fișierului unde începe subsecvența specificând numărul de ordine al octetului și al bitului din octet (numerotarea începe de la 0). De exemplu, pentru secvența dată mai jos, subsecvența căutată are lungimea 7 și începe de la octetul 2, bitul 5.

0	1	2	3	4
0111 0100	1001 1101	1000 0111	1111 0100	0110 0101

   Dacă există mai multe subsecvențe de aceeași lungime maximă, se va afișa primul offset.
   - Formatul apelului:
     `./t1 runs path=<file_path>`
   - Formatul rezultatului:
     i. Rezultat cu succes

Posibil rezultat cu succes
SUCCESS Length of the longest run: 7 Offset: 2 bytes + 5 bits

     ii. Rezultat cu eșec

Posibil rezultat pentru cale invalidă
ERROR Invalid file path

3. **Căutarea șabloanelor de biți** - care sunt date într-un fișier cu formatul descris mai jos.
   - **(20p)** Căutarea presupune contorizarea numărului de apariții a șablonului de biți specificat, prin suprapunerea șablonului peste conținutul secvenței analizate pornind de la prima poziție, apoi deplasând cu un bit, șamd. până la finalul secvenței.

   - Fișierul cu șabloane are un format bine definit, constând din METADATE + ȘABLOANE.
   - metadatele șabloanelor: se află la începutul fișierului și respectă structura:
     - VERSION: versiune de fișier de șabloane – 2 octeți
     - NO_OF_TEMPLATE_HEADERS: numărul de categorii de șabloane – 2 octeți
       a. Șabloanele de aceeași dimensiune formează o categorie
     - TEMPLATE_HEADERS: lista cu metadatele categoriilor NO_OF_TEMPLATE_HEADERS * sizeof(TEMPLATE_HEADER)
       a. TEMPLATE_HEADER: 2 + 2 + 4 octeți
          i. SIZE_OF_TEMPLATES: 2 octeți
             - Dimensiunea în biți a șabloanelor din această categorie
             - Dimensiunile suportate sunt: 8, 16, 24, 32 biți
          ii. NO_OF_TEMPLATES: 2 octeți
             - Numărul de șabloane din această categorie
          iii. OFFSET: 4 octeți
             - Poziția din fișier unde încep șabloanele din această categorie

   - șabloanele propriu-zise
     - șir de octeți conținând șabloanele din fiecare categorie
     - Șabloanele din categoriile consecutive nu sunt neaparat una după alta, astfel între două categorii consecutive pot exista octeți ce nu aparțin nici unei categorii. Este important să urmați OFFSET-ul pentru a afla unde începe următoarea categorie de șabloane.

- Formatul apelului:
  `./t1 template bits=<number_of_bits> template=<number_of_template> t_path=<template_file_path> path=<file_path>`
  sau
  `./t1 template bits=<number_of_bits> t_path=<template_file_path> path=<file_path>`

  (5p) Parametrul `bits` indică din ce categorie (categoria este dată prin specificarea dimensiunii șabloanelor din aceea categorie) dorim să selectăm șablonul cu numărul de ordine `number_of_template`
  Dacă lipeste numărul de ordine, se vor căuta toate șabloanele din categoria dată

  (5p) Dacă fișierul de șabloane nu are o structură validă, se afișează mesaj de eroare specificând primul motiv găsit dintre următoarele:
    - versiunea nu este corectă – valorile admise sunt între 12345 – 54321
    - categoriile de șabloane se suprapun
    - metadatele sunt incorecte

- Formatul rezultatului:
  i. Rezultat cu succes

Posibil rezultat cu succes
SUCCESS Occurences of template 3 of length 32: 10

Posibil rezultat cu succes pentru căutarea tuturor șabloanelor pe 16 biți
SUCCESS Number of templates in category: 4 Occurences of template 1 of length 16: 21 Occurences of template 2 of length 16: 24 Occurences of template 3 of length 16: 22 Occurences of template 4 of length 16: 19

  ii. Rezultat cu eșec

Posibil rezultat pentru cale invalidă
ERROR Invalid file path

Posibil rezultat pentru categorie invalidă
ERROR Invalid template category

Posibil rezultat pentru număr de șablon invalid
ERROR Invalid template number

Posibil rezultat pentru fișier de șabloane invalid
ERROR Invalid template file Wrong version / Overlaying categories / wrong metadata

4. **Listare cu filtrare**

Listarea conținutului directorului primit ca și argument – respectând criteriile de filtrare specificate. **(5p)** Numele fișierelor sau subdirectoarelor care trebuie listate vor apărea pe câte o linie separată, și trebuie să includă calea către directorul specificat ca și argument în *path*. Dacă nu se găsește nimic de afișat, apare doar mesajul SUCCES.

- Formatul apelului:
  `./t1 list [recursive] <filtering_options> path=<dir_path>`
- **(5p)** Dacă apare opțiunea *recursive*, căutarea va traversa întrega structură a subarborelui începând de la directorul specificat în *path*.
- Criterii de filtrare
  i. **(4p)** template=ok
     - listează fișierele care au structura validă de fișier de șabloane
     - Rezultat cu succes

Posibil rezultat cu succes
SUCCESS test_root/test_dir/file_name_1 test_root/test_dir/file_name_2 test_root/test_dir/subdir_1/file_name_1 test_root/test_dir/subdir_1/subdir_1_1/file_name_2 test_root/test_dir/subdir_2/file_name_3 ...

     - Rezultat cu eșec

Posibil rezultat pentru cale invalidă
ERROR Invalid directory path

  ii. **(5p)** hist_random=N
     - listează fișierele care sunt evaluate ca fiind aleatoare dpdv al testului histogramei
       a. Pentru valoarea specificată N (valorile suportate pentru N sunt 1, 2, 4, 8), frecvența de apariție a tuturor valorilor posibile pe N biți este aproximativ egală (diferența maximă admisă este de 1%).
  iii. **(3p)** name_contains=string
     - listează fișierele sau subdirectoarele care conțin în nume șirul de caractere *string*
  iv. **(3p)** size_greater=value
     - listează fișierele care au dimensiunea în octeți strict mai mare dacât valoarea specificată în argument

Vom testa câte un singur criteriu de filtrare la evaluare, dar opțional puteți integra cazul general, în care pot fi specificate mai multe criterii în aceeași comandă.

3. **Observații și recomandări**
   1. Pentru citirea corectă a opțiunilor si argumentelor și validarea acestora se acordă **15p**.
   2. Dacă programul nu respectă stilul de programare cerut, se poate aplica o penalizare de 10%
   3. Se vor folosi doar apeluri sistem pentru lucrul cu fișiere și directoare. Pentru citire și afișare se pot folosi funcțiile printf() și scanf(), perror() și funcții pentru gestiunea șirurilor cum ar fi sscanf() și snprintf().
   4. Pentru spargerea șirurilor în elemente, recomandăm utilizarea funcțiilor strtok() și strtok_r().