Indicații

- Testul conține 3 subiecte, durează 120 de minute și valorează 50 de puncte, cu alte 5 bonus.
- Se pot obține punctaje parțiale pentru un subiect numai dacă acest lucru este specificat.
- Pentru a fi punctată, o rezolvare **trebuie** să includă și metoda de verificare a funcționalității acesteia.
- Pentru subiectul 1, pentru afișarea la ecran este recomandată utilizarea macroului `PRINTF32`.
- Pentru subiectul 2, pentru afișarea la ecran este obligatorie utilizarea funcției de bibliotecă `printf()`.
- Ordinea de rezolvare a subiectelor este la alegerea voastră.
- Subiectele se rezolvă pe mașina virtuală de PCLP2.

Subiecte

Subiectul 1

a) [5p] Calculați numărul de biți zero consecutivi de la coada variabilei `zeros` folosind deplasări și comparații. Afișați rezultatul folosind macro-ul `PRINTF32`.

b) [5p] Efectuați operația ROT13 (rotire cu 13 poziții a unui caracter in intervalul a-z) asupra fiecărui caracter din variabila "vec" și afișați rezultatul folosind macro-ul `PRINTF32`.

c) [5p] Completați definiția structurii `endianess` astfel încât să conțină un câmp de tip char[8] numit `little` și un câmp de tip char[8] numit `big`. Afișați dimensiunea structurii în biți și bytes folosind macro-ul `PRINTF32`.

d) [5p] Vom considera reprezentarea "big-endian" a unui byte ca fiind ordinea în care bitul cel mai semnificativ este stocat primul, iar "little-endian" ca fiind ordinea în care bitul cel mai puțin semnificativ este stocat primul. Pentru fiecare caracter din variabila `vec`, calculați reprezentarea sa binară în format little-endian și big-endian, stocând rezultatele în câmpurile corespunzătoare ale structurii `endianess`. Fiecare structură `endianess` va fi stocată într-un tablou de structuri numit `endianness_vec`. Afișați rezultatele folosind macro-ul `PRINTF32`.

Subiectul 2

Un Generator de Numere Pseudo-Aleatoare LCG funcționează pe baza unei relații de recurență de forma: X_{n+1} = (a * X_n + b) mod m unde:

- X_n este numărul aleatoriu curent,
- a, b și m sunt parametrii generatorului.

Vom considera că X_0 este 0.

a) [5p] Alocați un buffer de 32 de numere de tip `int` prin intermediul funcției "malloc" și inițializați-l cu numere de la 0 la 31, apoi afișați-l.

b) [5p] Creați funcția "init_lcg" cu semnătura `void init_lcg(int, int, int)` care primește trei parametri: `a`, `b` și `m`, și inițializează un generator de numere pseudo-aleatoare folosind metoda LCG. Apelați această funcție cu a = 1103515245, b = 12345 și m = 2^{31} pentru a inițializa generatorul. Este la latitudinea voastră să decideți cum să stocați starea generatorului.

c) [5p] Creați o funcție "next_lcg" cu semnătura `int next_lcg(int)` care primește numărul curent și returnează următorul număr aleatoriu generat de LCG. Această funcție va folosi numerele a, b și m de la subpunctul anterior pentru a calcula următorul număr. Apelați această funcție cu numărul 1234 și afișați rezultatul.

d) [5p] Creați o funcție "map" cu semnătura `void map(int *, int, void (*)(int *))` care primește un buffer, lungimea acestuia și o altă funcție ca parametru. Această funcție va aplica funcția dată ca parametru fiecărui element din buffer. Apelați funcția "map" cu buffer-ul de 32 de octeți creat la punctul a) și funcția "next_lcg" pentru a transforma fiecare element al buffer-ului folosind generatorul LCG. Afișați conținutul buffer-ului după aplicarea funcției "map".

Subiectul 3

a) [5p] Scrieți un program în care să apelați funcția `print_message` din fișierul `msg.o`.

b) [5p] Inspectați executabilul "passcheck" și aflați parola corectă.

c) [5p] Inspectați executabilul “flawless” și găsiți o metodă de a afișa mesajul “Felicitari! Ai castigat!”.