01101001
01101111
01100011
01101100
01100001

Nume și grupă:

Introducerea în Organizarea Calculatoarelor și Limbaje de Asamblare

Model 2021-2022 (v2)

Timp de lucru: 120 de minute

1. Andrei vrea să invete mai bine cum se lucreaza cu bitii. Pentru asta si-a propus să rezolve cateva exercitii. Ajutati-l pe Andrei sa inteleaga mai bine cum se lucrează cu bitii.

a. Pentru o valoare întreagă definita pe 4 octeti afișați cei mai putin semnificativi 2 biti din al 2-lea cel mai semnificativ octet. Pentru testare folosiți variabila `num` deja definita în program. (**5 puncte**)

b. Pentru o valoare întreagă definita pe 4 octeti calculati numarul de biti de 1 de pe pozitiile pare ale numarului. Se considera ca primul bit se afla pe pozitia 0. Pentru testare folosiți variabila `num` deja definita în program. (**5 puncte**)

c. Pentru o valoare întreagă definita pe 4 octeti calculati numarul de grupuri de 3 biti consecutivi de 1. Pentru testare folosiți variabila `num` deja definita în program. (**5 puncte**)

Date de test pentru variabila `num` si rezultatele asteptate la fiecare punct se găsesc în fișierul `results.txt`.

2. **a.** Alocati pe stiva un vector de 20 de elemente de tip **byte** care sa fie initializate incremental pornind de la valoarea `'A'` (a[0] = 'A', a[1] = 'B', a[2] = 'C' etc.). Afisati vectorul pentru a demonstra corectitudinea. (**5 puncte**)

b. Implementați funcția `array_reverse` care inverseaza "in-place" un vector de **bytes** primit ca parametru. Semnatura functiei este **void array_reverse(char* arr, unsigned int length)**. Demonstrati faptul ca functia a fost implementata corect, apeland-o pe un exemplu la alegere. (**5 puncte**)

c. Implementați funcția `pow_array` care primeste urmatorii parametri: un pointer catre un vector de numere reprezentate pe un octet, lungimea acestui vector, si un pointer catre un vector de numere reprezentate pe 2 octeti. Functia calculeaza pentru fiecare element din primul vector patratul perfect si il stocheaza in cel de al 2-lea vector (b[i] = a[i]*a[i]). Puteti sa definiti cei 2 vectori in ce zona doriti. Demonstrati ca functia a fost implementata corect, apeland-o pe un exemplu la alegere. (**5 puncte**)

d. Inițializați tabloul de octeți byte_array cu valori byte astfel încât apelul macroului PRINT_HEX va afișa stringul "babadac". (**5 puncte**)

3. Dupa ce a aprofundat notiunile elementare de asamblare, Andrei isi doreste sa treaca la nivelul urmator si sa invete si concepte de securitate. Pentru asta si-a luat un tutorial de pe Internet despre "reverse engineering". Desi a inteles foarte bine ce l-ati invatat in trecut acum nu prea pricepe ce se intampla. Ca sa depaseasca acest impas, Andrei v-a adus un binar si va roaga sa il ajutati sa inteleaga cum poate sa ii descopere vulnerabilitatile si sa profite de ele.

a. Analizati binarul `attack.c` si descoperiti o cale de atac prin care sa apelati functia `invisible_func`. In caz de succes, va fi printat mesajul "Someone call the police!". (**10 puncte**)

Indiciu : Daca dorim sa folosim un input de 40 de caractere urmat de adresa 0xff292929 pentru binarul "attack", folosim comanda:

```
python -c 'print "A" * 40 + "\x29\x29\x29\xff"' | ./attack
```