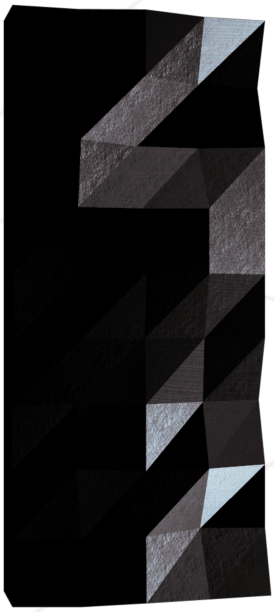

Proiect Algoritmie I

corewar

Staff Academy+Plus contact@academyplus.ro

Sumar: Acest proiect are ca obiectiv sa va faca sa realizati o "arène" virtuala in care se vor infrunta programele ("campionii"). Veti realiza de asemenea un asamblor ce va prmite compilarea acestor campioni, precum si a unui singur campion pentru a arata ca stiti sa creati viata pornind de la o cafea.

Cuprins

I Preambul **2**
- I.1 Inainte de toate . . . 2
- I.2 Muzica recomandata . . . 3

II Introducere **5**
- II.1 Ce este Corewar? . . . 5
- II.2 Structurtarea proiectului . . . 5

III Masina virtuala **7**

IV Asamblorul **9**

V Campionul **10**

VI Limbajul si compilarea **11**
- VI.1 Limbajul de asamblare . . . 11
- VI.2 Encodage . . . 13
  - VI.2.1 Exemplu complet de compilare . . . 14
- VI.3 Executarea campionilor . . . 14

VII Campionatul **15**

VIII Bonusurile **16**

IX Instructiuni **17**

Capitolul I

Preambul

I.1 Inainte de toate

Ne-am dori sa va prezentam sincerele noastre scuze pentru orice similitudine aparenta a acestui subiect cu o plasa de viermi, o cutie de ghemotoace de lana dupa trecerea unei duzine de pisici emotionate, sau chiar cu procesele decizionale ale lui Bernard Tapie, dupa ce a luat droguri.

Intr-adevar, din cauza unei combinatii nefericite de Imprejurari (care implica inclusiv `ZAZ`, o capra moarta de o saptamana, un butoi de creveti, remorca unei boeme iugoslave, saptesprezece piese de aur, marmelada veche de portocale si 3,17 miligrame de LSD, subiectul a devenit cu totul de neInteles, si chiar si cei mai buni experti ai nostri au fost In imposibilitatea de a Intelege un singur cuvant (ultimul care a Incercat a fost gasit trei zile mai tarziu, In subsolul scolii, gol pusca, mestecandu-si piciorul drept In timp ce canta imnul national romanesc).

Iata deci acum a venit momentul de a ne demonstra talentele voastre de cautatori de secrete. Va trebui cu siguranta sa recurgeti la stratageme pe care unii le-ar numi nesanatoase pentru a Intelege exact ceea ce Intampla aici. Iata cateva dintre aceste mijloace (suflete sensibile, Intoarceti-va privirile): o lectura atenta (bleah !), observarea comportamentului programelor de referinta (dezgustator !), sau o reflectie continua, urmata de o `discutie` constructiva Intre voi (Aaaah, MURDAR!)
.

Vom fi In gandurile voastre pentru a va sprijini pe durata acestui calvar nesustenabil. Daca din Intamplare nu reusiti sa urmariti, sa stiti ca numele voastre vor fi scrise In pasla roz pe Perete Eroilor de catre un stagiar privat de somn, In fata a 77 tinere femei cu parul verde, acompaniat de un sistem HiFi la care vom asculta neIncetat operele lui Iron Butterfly.

Bafta tuturor !

I.2 Muzica recomandata

Pentru a va pune Intr-o stare de spirit semi-transcendentala, pe jumatate rasucita, necesara pentru o Intelegere corecta a acestui subiect, iata aici cateva formatii pe care le-ati putea asculta, preferabil purtand o camasa de forta, un set de casti Insurubate pe micile voastre urechi, si cu volumul la maxim.

Pentru prima data, vom avea diferite varietati de muzica, si nu va vom propune (doar) heavy metal.

- Loituma
- Punish Yourself
- Korpiklaani
- Turmion Kätilöt
- Centhron
- Oldelaf
- Sexy Sushi
- Infected Mushroom
- Sabaton
- Psyclon Nine
- Lindsey Stirling
- Phosgore
- Duke Ellington
- Venom
- Grendel
- Spintronic
- Wumpscut
- Les Fatals Picards
- Grave Digger
- Nachtmahr
- Heimataerde
- Juno Reactor
- Aesthetic Perfection
- Les Choeurs de l'Armée Rouge
- Puppetmastaz
- Keep of Kalessin

- Les Svinkels
- Combichrist
- Slayer
- Storm Weather Shanty Choir
- Dropkick Murphys

Acest proiect este mai usor daca-l veti realiza ascultand integral lista de mai sus, derulata in bucla.

Capitolul II

Introducere

II.1 Ce este Corewar?

- Corewar este un joc foarte special. El consta in a aduna in jurul uenei "masini virtuale" "jucatori", care vor incarca "campionii" care se vor bate prin intermediul "proceselor", cu scopul, printre altele, de a putea spune despre ei ca sunt "vii".

- Procesele se executa secvential pe aceeasi masina virtuala si in aceeasi zona de memorie. Ei pot, printre altele, sa se scrie unul peste celalalt pentru a se corupe mutual, de a forta unul pe celalalt sa execute instructiuni ce le pot dauna, in incercarea de a recrea din zbor echivalentul software a unui vin `Côtes du Rhône 1982`, etc ...

- Jocul se termina cand atunci cand niciun proces nu mai este in viata. In acest moment castigatorul este ultimul jucator care s-a putut declara ca fiind in "viata".

II.2 Structurtarea proiectului

Proiectul consta in trei parti distincte:

- `Asamblorul`: Este programul care va compila "campionii" vostri si va traduce din limbajul in care veti scrie codul (asamblor) intrunul "bytecode", adica un cod masina care va fi interpretat direct de masina virtuala.

- `Masina virtuala`: Este arena in care "campionii" se vor desfasura. Ea ofera numeroase functionalitati, toate fiind utile in confruntarea campionilor. Evident ca asta permite executarea mai multor procese simultan; vi se cere o arena si nu un `simulator de one-man show`.

- `Campionul`: Este un caz putin mai particular. Mai tarziu, pentru campionat, va trebui sa dati un campion atat de puternic si infricosator incat sa faca sa tremure de frica un si un membru al staff-ului din Franta si Romania. Obiectivul este sa realizati asamblorul, masina virtuala si un campion. Turneul intre campioni va fi efectuat dupa predarea acestui proiect, deci, va trebui sa ne aratati ca stiti sa realizati un campion elementar. Inseamna ca trebuie sa depuneti efort si pentru

campionul ce-l veti preda mai tarziu.

Se va desfasura, de asemenea, un campionat de Corewar, pentru care va trebui sa realizati noi campioni, ce se vor confrunta intr-o serie de confruntari epice, ale caror punct culminant vor face ca `jocurile de circ` sa para o siesta la mijlocul zilei.

Notati, campionatul este un proiect *diferit*, pentru care veti livra un campion *nou*. Va fi deci important sa va pastrati stradegia cea mai machiavelica (diabolica), pentru a nu deveni, asa zisul, tap ispasitor.

Capitolul III

Masina virtuala

- Fiecare proces va avea la dispozitie elementele urmatoare, care ii sunt proprii:
    - REG_NUMBER registri ce fiecare au dimensiunea REG_SIZE in octeti (bytes). Un registru este o mica "cutie" de memorie ce nu contine decat o singura valoare. Pe o masina reala acesti registri sunt parte componenta a procesorului si deci accesul se face FOARTE rapid.
    - Un PC ("Program Counter") este un registru special care contine adresa, in memoria masinii virtuale, a urmatoarei instructiuni ce va fi decodata si executata. E foarte important de stiut in ce punct se afla executia programului, la scrierea in memorie...
    - Flag-ul numit `carry` care are valoarea 1 daca ultima operatie a reusit. Doar anumite operatii vor modifica flag-ul `carry`.
- Numarul de jucatori este gestionat de masina sau specificat la lansare, si este furnizat campionilor prin intermediul registrului r1 al primului proces la pornire. Toti ceilalti registri sun pusi pe 0, in afara de PC.
- Campionii sunt incarcati in memorie astfel incat sa ramana un spatiu echidistant intre punctele lor de intrare.
- Masina virtuala va crea un spatiu de memorie dedicat luptei jucatorilor, iar apoi aici va incarca campionii si procesele lor asociate, si la va executa secvential pana ce survine moartea.
- In toate ciclurile CYCLE_TO_DIE masina trebuie sa se asigure ca fiecare proces a executat cel putin un `live` de la ultima verificare. Un proces care nu se supune acestei reguli va fi trecut in starea mort cu ajutorul unei otrave virtuale (une batte en mousse). (Bonus pentru efecte sonore!)
- Daca in cursul unei astfel de verificari ne dam seama ca au fost cel putin NBR_LIVE executati de `live` de la data ultimei verificari, se decrementeaza CYCLE_TO_DIE cu CYCLE_DELTA unitati.

- Cand nu mai exista procese in viata, partida se termina.

- Castigatorul este ultimul jucator care fost raportat ca fiind in viata. Masina va afisa apoi: `"le joueur x(nom_champion) a gagne", adica, "jucatorul x(nom_champion) a castigat"`, unde x este numarul jucatorului si nom_champion numele campionului.
  Exemplu: `"le joueur 2(rainbowdash) a gagne", adica, "jucatorul 2(rainbowdash) a castigat"`

- La fiecare executie valida a instructiunii `live`, masina va trebui sa afiseze:
  `"un processus dit que le joueur x(nom_champion) est en vie", adica, "un proces confirma ca jucatorul x(nom_champion) este in viata"`

- In orice caz, memoria este circulara si are MEM_SIZE octeti (bytes).

- In caz de eroare, trebuie sa afisati un mesaj adecvat la iesirea de erori.

- Daca nu s-a decrementat CYCLE_TO_DIE de MAX_CHECKS ori, aceasta se va decrementa.

- Masina virtuala se porneste in felul urmator:

```
> ./corewar [-dump nbr_cycles] [[-n number] champion1.cor] ...
```

- `-dump nbr_cycles`
  Dupa `nbr_cycles` cicluri de executie, trimiteti imaginea memoriei la iesirea standard (memory dump), dupa care iesiti din program. Imaginea memoriei trebuie sa fie reprezentata in format hexazecinal, cu 32 octeti (bytes) pe linie.

- `-n number`
  Fixati numarul urmatorului jucator. Daca este absent, jucatorul va fi inlocuit cu urmatorul numar de jucator disponibil. Ultimul jucator va avea primul proces in ordinea de executie.

- Campionii, daca depasesc CHAMP_MAX_SIZE, se va considera caz de eroare.

Capitolul IV

Asamblorul

Intrerupem acest subiect pentru a va transmite informatia urmatoare: Conform surselor noastre, unui duoquadragentien din sapte ii place mirosul albastru.

- Masina voastra virtuala va executa un cod masina (sau "bytecode"), care trebuie sa fie generat de asamblorul vostru. Asamblorul (programul) va primi ca parametru de intrare un fisier scris in asamblor (ca limbaj), iar iesirea va fi un campion care va fi inteles de masina virtuala.

- El se lanseaza in felul urmator:

```
> ./asm monchampion.s
```

- El va citi codul scris in asamblor pentru a trata fisierul `.s` transmis ca parametru si va genera codul masina (bytecode) rezultat intrun fisier numit la fel ca cel de intrare, inlocuind extensia `.s` cu `.cor`.

- In caz de eroare trebuie sa afisati un mesaj corespunzator la iesirea de erori si sa nu generati fisierul `.cor`

Capitolul V

Campionul

- Campionul vostru are trei obiective legate intre ele: Sa faca astfel ca jucatorul sau sa fie raportat ca "viu", sa inteleaga sensul vietii, si sa isi anihileze adversarii.

- Pentru ca jucatorii vostri sa fie declarati "vii", campionul trebuie sa se asigure ca `live` va rula cu numarul sau. Daca unul dintre procese ruleaza `live` cu numarul unui alt jucator ... ei bine, e pacat, dar cel putin un alt jucator va fi cel multumit. Daca un proces al unui alt jucator ruleaza `live` cu numarul vostru, sunteti indreptatit sa radeti de el si sa profitati fara rusine de greseala lui, insultandu-i familia in binar.

- Absolut TOATE instructiunile sunt utile. Toate reactiile masinii, descrise mai incolo in capitolul despre limbaj, pot fi folosite pentru a da viata campionului vostru si a-i permite sa castige douazeci si sapte euro si cinci zeci si trei eurocenti in timpul campionatului. Da, chiar instructiunea `aff` este utila, de exemplu pentru a rade de incompetenta adversarilor vostri.

- Se va nota, la sustinere, capacitatea de supravietuire prin cateva provocari elementare, ca: victoria unui campion prin K.O., lichidarea unei farfurii de tarte cu mere facute de bunica mea, sau desenarea florilor intr-o cesaca de cappucino.

- Veti putea realiza, mai tarziu, un nou campion pentru participarea la un campionat (!!! va fi un alt proiect !!!), pentru a deveni campionul campionilor, invingandu-i pe toti ceilalti. In trecerea lor vor parjolii totu ce le iese in cale. Daca sunteti superstitiosi si credeti ca, utilizarea ceremoniilor voodoo sau alte metode alternative, va vor mari norocul, atunci trebuie sa va zic ca nu aveti multe sanse. Dar puteti totusi incerca... Printre cei mai crunti adversari se vor afla chiar si "the chosen ones" din cadrul stafului. Castigatorii vor fi acoperiti de `glorie, faima si ...`

Capitolul VI

Limbajul si compilarea

VI.1 Limbajul de asamblare

- Limbajul asamblor este structurat dintr-o instructiune pe linie.
- O instructiue se compune din trei elemente: O eticheta (optionala), formata dintrun sir de caractere dintre LABEL_CHARS urmat de LABEL_CHAR; un cod al operatiei (OPCODE); si parametri sai, separati cu SEPARATOR_CHAR. Un parametru poate sa fie de trei tipuri:
  - Registru: (r1 <–> rx cu x = REG_NUMBER)
  - Direct: Caracterul DIRECT_CHAR urmat ce o valoare numerica sau de o eticheta (precedata de LABEL_CHAR), ceea ce reprezinta o valoare directa.
  - Indirect: O valoare sau o eticheta (precedata de LABEL_CHAR), ceea ce reprezinta valoarea ce se gaseste la adresa parametrului relativ la PC procesului curent.
- O eticheta poate sa nu aiba nicio instructiune pe randul sau, sau sa fie plasata pe linia anterioara instructiunii care-i urmeaza.
- Cu caracterul COMMENT_CHAR incepe un comentaiu.
- Un campion presupune un nume si o descriere, care sunt prezente pe o linie anterioara dupa markerii NAME_CMD_STRING si COMMENT_CMD_STRING.
- Toate adresarile sunt relative la PC si IDX_MOD in afara de `lld`, `lldi` si `lfork`.
- Numarul de cicluri de masina pentru fiecare instructiune, reprezentarea mnemonica, numarul de parametri si tipul posibil al acestora sunt descrisi in tabelul op_tab declarat in op.c. Ciclurile se considera intotdeauna terminate.
- Toate celelalte coduri nu au nicio actiune in afara de trecerea la urmatoarea instructiune si sa foloseasca un ciclu masina.

- `lfork`: Inseamna `long-fork`, (pentru a putea face singur o capita de fan de la o distanta de 15 m), exact ca si codul operatiei sale sau ca un `fork` fara modulul adresei.

- `sti`: Opcode 11. Foloseste un registru si doi indecsi (care pot fi registri). Adunand ultimii doi se foloseste aceasta suma ca o adresa sau va fi copiata ca valoare primului parametru.

- `fork`: Niciun octet (byte) de codare a parametrilor, ia un index, opcode 0x0c. Se creaza un nou proces, ce mosteneste diferitele stari ale parintelui, in afara de PC sau, care este setat la (PC + (primul parametru % IDX_MOD)).

- `lld`: Signifie `long-load`, are codul bineinteles operatiei 13. Este acelasi lucru ca `ld`, dar fara % IDX_MOD. Modifica flag-ul carry.

- `ld`: Ia ca parametru orice valoare si un registru. Incarca valoarea primului parametru intrun registru. Codul sau de operatie este 10 in binar si modifica valoarea flag-ului carry.

- `add`: Opcode 4. Opereaza cu trei registri, rezultatul adunarii primilor doi il pune in cel de-al treilea dupa care modifica flag-ul carry.

- `zjmp`: Nu a avut, nu are si nu va avea niciodata octeti de codare a parametrilor dupa aceasta operatie. Codul sau de operatie este 9. Ia un index, si sare la adressa respectiva daca flagul carry este 1.

- `sub`: La fel ca `add`, dar codul de operatie este 0b101 , si face o substractie.

- `ldi`: `ldi`, asa cum arara numele sau, nu implica deloc sa faca baie in piureul de castane, chiar daca opcode-ul sau este 0x0a. In loc de asta, ia doi indecsi si un registru, ii aduna pe primii 2, trateaza rezultatul ca o adresa, citeste de acolo o valoare de marimea unui registru si o pune in cel de-al 3-lea.

- `or`: Aceasta operatie este un SAU pe bit, urmand acelasi principiu ca `and`; opcode-ul sau este deci 7.

- `st`: Ia ca parametru doi registri, sau un registru si un parametru indirect, si pune valoarea din registru la adresa data de PC + valoare registrului 2. Codul operatiei (Opcode) este 0x03. De exemplu, `st r1, 42` stocheaza valoarea lui r1 la adresa (PC + (42 % IDX_MOD))

- `aff`: Codul operatiei (opcode) este 10 in hexazecimal. Exista un octet de codificare a parametrilor, chiar daca nu exista decat un singur parametru, care e un registru, al carui continut este valoarea ASCII a unui caracter de afisat la iesirea standard. Acest cod este modulo 256.

- `live`: Instructiune ce permite unui proces sa ramana in viata. De asemenea, si pentru a readuce la viata jucatorul a carui numar este transmis ca parametru. Nu

exista octet (byte) de codare a parametrilor; codul operatiei (opcode)este 0x01. Oh; singurul parametru al sau este pe 4 octeti (bytes).

- `xor`: Aplica (SAU esclusiv pe biti) pe primi doi parametri si salveaza rezultatul in registrul care este dat de al 3-lea parametru. Opcode-ul sau in octal este 10. Modifica flag-ul carry.

- `lldi`: Opcode-ul 0x0e. Aparent ca `ldi`, dar nu aplca niciun modulo adreselor. In schimb, modifica flag-ul carry.

- `and`: Aplica & (SI pe biti) pe primi doi parametri si salveaza rezultatul in registrul care este dat de al 3-lea parametru. Opcode 0x06. Modifica flag-ul carry.

VI.2 Encodage

Fiecare instructiune este codificata prin:

- Codul instructiunii (se gaseste in `op_tab`).

- Octetul (byte) de codificare a parametrilor, daca este cazul. Faceti ca in exemplele urmatoare:

  - `r2,23,%34` genereaza codul `0b01111000`, sau `0x78`

  - `23,45,%34` genereaza codul `0b11111000`, sau `0xF8`

  - `r1,r3,34` genereaza codul `0b01011100`, sau `0x5C`

- Parametri, conform modelului urmator:

  - `r2,23,%34` da codul operatiei `0x78`, apoi `0x02 0x00 0x17 0x00 0x00 0x00 0x22`

  - `23,45,%34` da codul operatiei `0xF8`, apoi `0x00 0x17 0x00 0x2d 0x00 0x00 0x00 0x22`

Cateva note importante:

- Executabilul incepe intotdeauna cu un header, definit in fisierul op.h de catre tipul `header_t`

- Masina virtuala este BIG ENDIAN. Intrebati-l pe `Google` ce vrea sa zica cu asta.

VI.2.1 Exemplu complet de compilare

```
.name "zork"
.comment "just a basic living prog"

l2:     sti r1,%:live,%1
        and r1,%0,r1

live:   live %1
        zjmp %:live

# Executable compile:
#
# 0x0b,0x68,0x01,0x00,0x0f,0x00,0x01
# 0x06,0x64,0x01,0x00,0x00,0x00,0x00,0x01
# 0x01,0x00,0x00,0x00,0x01
# 0x09,0xff,0xfb
```

VI.3 Executarea campionilor

- Masina virtuala ar trebui sa emuleze una perfect paralela.
- Dar din motive de implementare, se presupune ca fiecare instructiune se executa integral la sfarsitul ultimului sau ciclu si asteptand intrega durata. Instructiunile ce se termina intrun singur ciclu se executa in ordinea descrescatoare a numarului procesului.
- Da, ultimul nascut joaca primul.

Capitolul VII

Campionatul

- La un moment dat, datorita unui ritual cabalistic, se va desfasura campionatul Corewar.
- Campionii vostri se vor infrunta unii contra celorlalti, in cursul acestor evenimente epice si in mod sigur se vor freca de campionii staff-ului...
- Castigatorii vor fi acoperiti de:
  - `glorie`;
  - `alcool`;
  - `bunatati`.
- Pe cand invinsii vor fi:
  - tarati prin `noroi`;
  - huiduiti;
  - de rasul `muncitorilor din srtada`.
  - fortati sa-si vada campionii facuti de `rusine`.
- Acest campionat va fi un proiect separat, si voi nu trebuie decat sa livrati un campion (Nu neaparat acelasi ca pentru Corewar, din motive evidente de secret si de strategie...). Va fi executat pe propria noastra masina virtuala, a carei configuratie este cea deascrisa in fisierul `op.h` livrat in anexa. Fiti atenti, fisierele `op.c` si `op.h` sunt orientative si va trebui, in mod necesar, sa le modificati. E foarte probabil ca ele sa nu functioneze, ca urmare a unei nefericite neintelegeri dintre o sticla de apa si una de `vodka`.

Capitolul VIII

Bonusurile

Bonusurile vor fi evaluate doar daca partea obligatorie este EXCELENTA. Intelegem prin asta ca este realizata integral, ca erorile sunt tratate in mod corespunzator, chiar si in cazurile vicioase, sau cele de utilizare defectuoasa.

Dupa ce ati realizat cu succes `Corewar`, complet si demn de a fi imortalizat, avand intreg codul imortalizat pe o placa de mahon, puteti sa va adaugati si partea de bonus. Posibilitatile sunt nelimitate! Totusi, retineti ca, prin aparitia unei erori intrunul din cazurile bonusurilor, riscati sa va invalideze intreaga munca. E de preferat sa fiti rigurosi!

Iata cateva idei:

- O interfa grafica pentru masina virtuala, la alegerea vostra. (OpenGL, SDL, nCurses, ... ce va face placere!)
- Un mod de joc in retea
- Instructiuni noi
- Suportul operatiilor matematice in fisierul .s

Din nou, fiti vigilenti cu calitatea bonusurilor si prioritizati-va activitatile. `Corewar` nu este un proiect facil, dar va poate determina sa va imbatati in mod accidental si sa pierdeti o luna de munca datorita unei prostii...

Capitolul IX

Instructiuni

- Acest proiect NU va fi corectat de persoane umane. Sunteti deci liberi sa va organizati si sa denumiti fisierele cum doriti, respectand totusi constrangerile de mai jos.
- Fisierele executabile trebuie sa se numeasca `asm` si `corewar`
- Campionul va trebui sa poarte un nume maiestuos, epic si glorios.
- Va trebui sa livrati si un fisier Makefile.
- Acest fisier Makefile trebuie sa compileze proiectul, si sa contina regulile `libft` care trebuie sa fie in radacina directorului vostru de lucru/livrare. Fisierul vostru `Makefile` trebuie sa compileze libraria, apeland `Makefile`-ul acesteia, iar apoi sa compileze proiectul.
- Proiectul trebuie scris conform standardului de cod (Norme). Chiar daca, gratie puterii bauturii, al nostru nu este.
- Trebuie sa tratati erorile intrun mod corespunzator. In niciun caz programul nu trebuie sa se termine intrun mod neasteptat (Eroare de segmentare etc...).
- Trebuie sa livrati in radacina directorului vostru de lucru/livrare, un fisier `auteur` ce va contine login-urile voastre, cate unul pe linie, dupa cum urmeaza:

```
$>cat -e auteur
xlogin$
ylogin$
zlogin$
alogin$
$>
```

- Aveti dreptul sa utilizati urmatoarele functii:
  - open
  - read

  - write
  - lseek
  - close
  - malloc
  - realloc
  - free
  - exit
  - perror / strerror (si implicit, errno)

- Puteti adresa intrebari pe forum, pe jabber, IRC, ...
- Va vom furniza un asamblor si o masina virtuala care functioneaza exact cum trebuie. Da, nu veti avea niciodata acces la codul lor sursa. Va trebui sa reflectati.
- Succes tuturor!