**ASC-TEMA1**

**Realizarea temei**

**Punctul 1**

Pentru a stabili dacă p-ul citit  în  input este prim vor fi tratate 3 cazuri:

* p<2
* p=2
* p>2

Dacă p este neprim, se va afișa acest aspect și se va ieși din program

Alfel, se va trece la calcularea generatorului prin parcurgea elemetelor de la 2 la p-1(posibili generatori ai grupului multiplicativ ( Zp\*,\*)

Pentru fiecare element  voi calcula (element^k)%p, valoarea lui k fiind incrementată cu 1 până când rezultatul este 1. La întâlnirea lui 1, se va verifică dacă k este egal cu p, adică dacă toate elemetele din Zp\* au fost întâlnite. În acest caz, se va afișa generatorul(=element) și se va trece la punctul 2. Altfel, se va trece la elementul următor și se va repeta procesul până la întâlnirea lui 1.

**Punctul 2**

Se va citi direct mesajul text și i se va calcula lungimea. Pentru a realiza criptarea mesajului citit se va fixa un alfabet de lucru în zona .data care va fi utilizat pentru  a obține poziția fiecărui caracter din șirul dat în alfabet(de ex.:A-poziția 0, B-poziția 1,etc.) prin scăderea codului ASCII a primului caracter din alfabet(A,65) din codul ASCII al caracterului curent. După ce este obținută poziția în alfabet a caracterului curent, se va realiza operația (generatorpozitie\_in\_alfabet)%p pentru a obține poziția în alfabet a caracterului din mesajul criptat.

**Punctul 3**

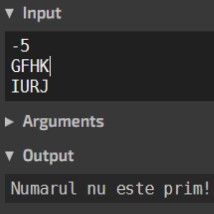
Se va citi direct mesajul text, caracter cu caracter  până la întâlnirea lui 0 și, concomitent cu citirea fiecărui caracter se va obține și poziția sa în alfabet prin același procedeu ca la punctul 2. În acest moment vom executa operația inversă celei de la puncul 2, adică , cunoscând faptul că generatorpozitie\_in\_alfabet\_mesaj\_initial%p = pozitie\_in\_alfabet\_mesaj\_criptat valoarea lui pozitie\_in\_alfabet\_mesaj\_initial va fi determinată astfel:

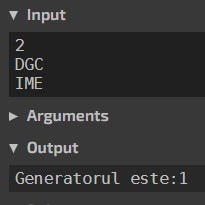
Pentru p=7, generator=3, iau pozitie\_in\_alfabet\_mesaj\_criptat=6 (poziție ce corespunde lui G)

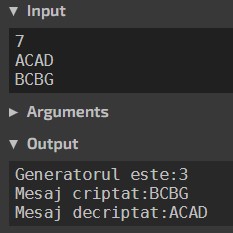
Atunci  3pozitie\_in\_alfabet\_mesaj\_initial=6,iar  pentru a putea executa această operație pe mulțimea nr. naturale îl vom aduna pe p (adică 7) la 6 până când se va obține o putere a lui 3 (operația inversă a lui %p). Deci  se va folosi o variabilă k care  reține exponentul 3pozitie\_in\_alfabet\_mesaj\_initial=6+k\*p, iar k va fi valoarea căutată(pozitie\_in\_alfabet\_mesaj\_initial).

Adică: 3pozitie\_in\_alfabet\_mesaj\_initial =6+3\*7, deci pozitie\_in\_alfabet\_mesaj\_initial=3 (deci litera D).

**Exemple**







**Cod**

.data

p:.space 4

negativ:.asciiz "Numarul nu este prim!"

sp:.asciiz " "

nl:.asciiz "\n"

criptat:.asciiz "Mesaj criptat:"

decriptat:.asciiz "Mesaj decriptat:"

alfabet:.asciiz "ABCDEFGHIJKLMNOPQRSTUVWXYZ"

generator:.asciiz "Generatorul este:"

mesaj\_citit:.space 100

mesaj\_criptat:.space 100

gen1:.asciiz "Generatorul este:1"

.text

main:

li $v0,5 #read int

syscall

move $t0,$v0 #obtin elemetul citit in $t0

sw $t0,p #salvez elementul citit p in $t0

#vom verifica daca p este sau nu prim

li $t1,2

blt $t0,$t1,iesire #daca p<2,se va merge la instructiunea'iesire',unde se va afisa mesajul negativ si programul se va opri din executie

beq $t0,$t1,generatorul\_1 #altfel, se va verifica daca p=2,iar daca este, se va afisa faptul ca generatorul este 1

#altfel, daca nu sunt indeplinite cazurile anterioare,se va trece la verificarea primalitatii pt numere>2

#prin parcurgerea divizorilor lui p de la 2 la p/2

div $t0,$t1 #il impart pe $t0(=p) la $t1(=2)

mflo $s0 #retin catul impartirii anterioare in lo

loop1:

bgt $t1,$s0,calculez\_generator #in cazul in care $t1>$s0(=p/2),voi executa instructiunea 'calculez\_generator'

div $t0,$t1 #il impart pe $t0(=p) la $t1

mfhi $s1 #retin restul impartirii anterioare in hi

beq $s1,$0,afis\_negativ #daca restul este 0, inseamna ca p are divizori proprii si i se va anunta neprimalitatea prin 'afis\_negativ'

addi $t1,1 #alfel, vom continua cu cautarea altor posibili divizori

j loop1 #revenim la loop1

afis\_negativ:

la $a0,negativ #incarcam in $a0 adresa mesajului de afisat

li $v0,4 #print string

syscall

li $v0,10 #exit

syscall

calculez\_generator:

li $t1,2 #il reinitializez pe $t1 cu 2 pentru urmatoarea parcurgere, in acest caz $t1 reprezinta valoarea generatorului

li $t2,1 #in t2 vom retine valoarea lui $t1^k

li $t4,1 #avem nevoie de valoarea lui 1 pt a-l putea compara pe $t2 cu 1

li $t5,1 #registrul $t5 este contorul care verifica daca au fost parcurse el lui Zp\*(p-1 elem) in momentul in care $t2 devine 1

loop2:

beq $t1,$t0,punctul\_2

mul $t3,$t1,$t2 #$t3=$t1\*$t2

div $t3,$t0 #il impart pe $t3 la $t0

mfhi $s0 #retin in $s0, restul operatiei anterioare

move $t2,$s0 #ii atribuim lui $t2 valoarea din $s0

addi $t5,1 #actualizam contorul

beq $t2,$t4,verif\_contor

j loop2

verif\_contor:

beq $t5,$t0,afis\_generator

addi $t1,1

li $t2,1

li $t5,1

j loop2

afis\_generator:

la $a0,generator

li $v0,4 #afisam mesajul "Generatorul este:"

syscall

move $a0,$t1

li $v0,1 #afisam generatorul

syscall

la $a0,nl

li $v0,4 #afisam linie noua

syscall

j punctul\_2

punctul\_2:

#criptare mesaj\_citit

la $a0,mesaj\_citit #in $a0 incarcam zona de mem unde vom stoca sirul de caractere citit

li $a1,99 #in $a1 vom stoca lungimea maxima a sirului citit

li $v0,8 #citim de la tastatura siriul mesaj\_citit

syscall

li $t2,0 #t2 sare din 1 in 1 pe locatiile de memorie

lb $t3,mesaj\_citit($t2) #$t3 e byte-ul curent din sir

li $t4,65 #retinem in $t4 codul ascii al primului caracter din alfabetul englez

la $a0,criptat

li $v0,4 #vom afisa mesajul "Mesaj criptat:" retinut in criptat

syscall

calculez\_lungime\_sir:

beqz $t3,initializari #daca $t3(byte-ul curent) este 0

addi $t2,1

lb $t3,mesaj\_citit($t2)

j calculez\_lungime\_sir

initializari:

addi $t2,-1

li $t9,0

lb $t3,mesaj\_citit($t9)

li $t4,65 #retinem in $t4 codul ascii al primului caracter din alfabetul englez

loop3:

beq $t9,$t2,punctul\_3 #daca $t3(byte-ul curent) este 0

sub $t5,$t3,$t4 #vom retine in $t5 pozitia pe care se afla fiecare caracter din mesaj\_citit in alfabet($t5=$t3-$t4)

#acum trebuie sa il ridic pe $t1(generatorul la puterea $t5,iar rezultatul va fi mod p)

li $t6,1 #in t6 vom retine indicele cu care parcurg exponentul $t5 la care il ridic pe generator

li $t7,1 #in t7 se va retine rezultatul pasului anterior a inmultirii

loop4:

bgt $t6,$t5,afisez\_intreg

mul $t8,$t7,$t1

div $t8,$t0

mfhi $s0

move $t7,$s0

addi $t6,1

j loop4

afisez\_intreg:

lb $a0,alfabet($t7)

li $v0,11

syscall

addi $t9,1

lb $t3,mesaj\_citit($t9)

j loop3

punctul\_3:

la $a0,nl

li $v0,4

syscall

#decriptare mesaj criptat

la $a0,mesaj\_criptat #in $a0 incarcam zona de mem unde vom stoca sirul de caractere citit

li $a1,99 #in $a1 vom stoca lungimea maxima a sirului citit

li $v0,8 #citim de la tastatura siriul mesaj\_citit

syscall

li $t2,0 #t2 sare din 1 in 1 pe locatiile de memorie

lb $t3,mesaj\_criptat($t2) #$t3 e byte-ul curent din sir

li $t4,65 #retinem in $t4 codul ascii al primului caracter(i.e. 'A') din alfabet

la $a0,decriptat

li $v0,4 #vom afisa mesajul "Mesaj decriptat:" retinut in decriptat

syscall

loop5:

beqz $t3,exit #daca $t3(byte-ul curent) este 0

sub $t5,$t3,$t4 #vom retine in $t5 pozitia pe care se afla fiecare caracter din mesaj\_citit in alfabet($t5=$t3-$t4)

li $t6,0 #contorul prin care il voi afla pe x((generator^x)mod p=y,y fiind pozitia din alfabet a unui elem din sirul criptat)

li $t7,1 #retinem 1 in $t7 pt a-l folosi la comapari ulterioare

loop6:

div $t5,$t1

mflo $s0 #cat

mfhi $s1 #rest

bne $s1,$0,verif\_cat #daca restul e diferit de 0, atunci trec la verif\_cat

move $t5,$s0 #actualizam valoarea deimpartitului cu catul actual

addi $t6,1 #actualizam valoarea contorului

j loop6

verif\_cat:

bgt $t5,$t7,adun\_p #daca deimpartitul e inca mai mare ca 1,atunci trec la adun\_p

lb $a0,alfabet($t6)

li $v0,11

syscall

addi $t2,1

lb $t3,mesaj\_criptat($t2)

j loop5

adun\_p:

add $t5,$t5,$t0

j loop6

exit:

li $v0,10

syscall

generatorul\_1:

la $a0,gen1

li $v0,4

syscall

li $v0,10 #exit

syscall

iesire:

la $a0,negativ

li $v0,4 #anunt neprimalitatea lui p prin mesajul negativ

syscall

li $v0,10 #exit

syscall