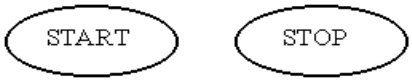
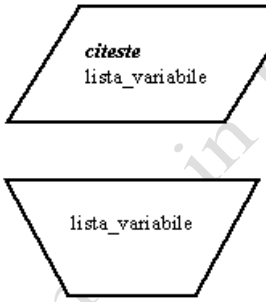
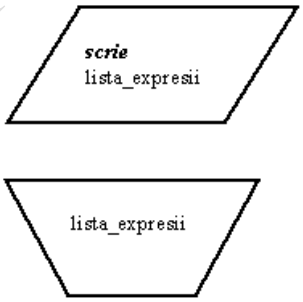
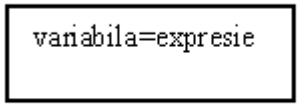
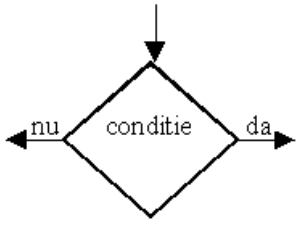
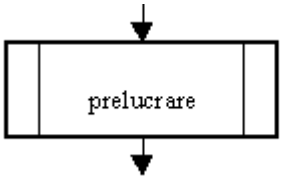
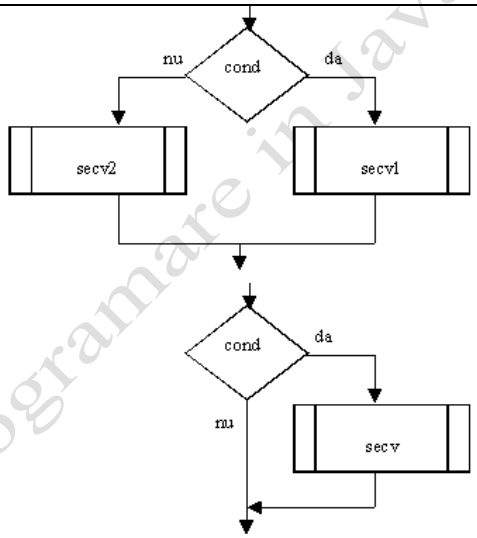
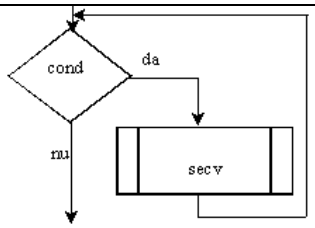
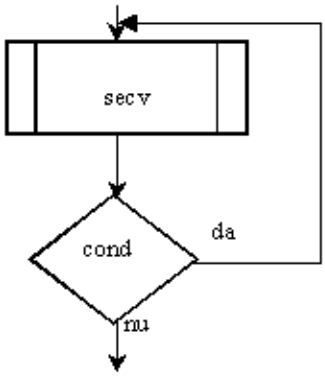
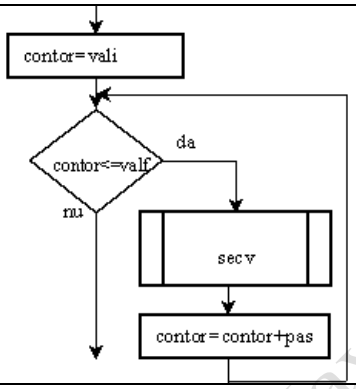
In **schemele logice** se folosesc blocurile indicate in **Tabelul 2.1**.

Blocul de inceput/sfarsit - orice schema logica incepe cu un bloc de inceput (start) si se termina cu blocul de sfarsit (stop)	
Blocul de citire (doua variante) - se citesc de la dispozitivul de intrare valorile variabilelor specificate in lista_variabile (separate prin virgula); se foloseşte la introducerea datelor (citire)	
Blocul de scriere (doua variante) - se scriu la dispozitivul de iesire valorile obtinute in urma evaluarii expresiilor din lista (separate prin virgula); se foloseşte la afişarea rezultatelor (scriere)	
Blocul de atribuire - se evalueaza expresia, iar valoarea obtinuta este memorata in variabila, vechea valoare pierzandu-se; expresia contine operatori (+ - * /), operanzi (variabile, constante) si ( ); variabila poate apare si in expresie (ex: a=a+5)	
Blocul de decizie - se evalueaza conditia: daca este adevarata se continua cu prelucrarea indicata de ramura ***da***, altfel se continua cu prelucrarea indicata de ramura ***nu***; conditia poate contine operatori relationali: < > <= >= == != operatori logici: *si*, *sau* etc	

Tabelul 2.1 Blocuri folosite în schemele logice

Structurile de control de baza impreuna cu cele derivate sunt prezentate in **tabelul 2.2.**

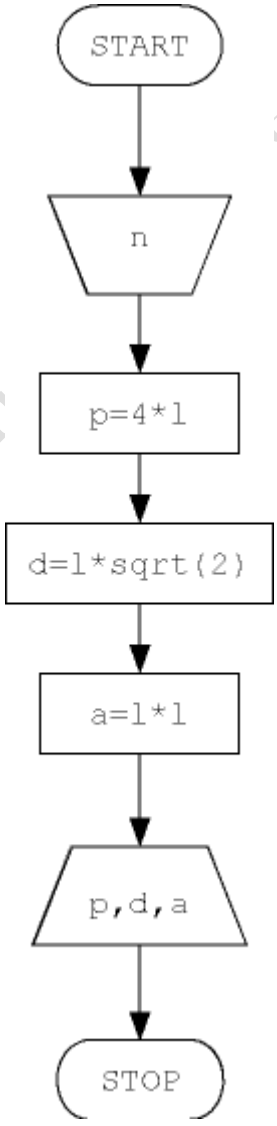
Secventa - indica executia succesiva a operatiilor de baza si a structurilor de control in ordinea in care apar in schema logica; in general, orice schema logica cuprinde secventele: de initializare a variabilelor, prelucrari, tiparirea rezultatelor	
Selectia - functie de valoarea de adevar a conditiei, se executa una din secvente, dupa care se trece la prelucrarea urmatoare; cele doua ramuri se exclud mutual; este posibil ca una din ramuri sa fie vida	
Ciclul cu test initial - cat timp conditia este adevarata, secventa (corpul ciclului) se executa ciclic; daca la prima evaluare a conditiei, aceasta este falsa, corpul nu se executa niciodata.	
Ciclul cu test final - conditia se evalueaza dupa executia secventei (corpul ciclului) care se executa cel putin o data; se revine la executia secventei, daca este adevarata conditia.	

Ciclul cu contor - ciclarea este controlata de o variabila numita generic *contor*; se initializeaza cu o valoare initiala, *vali*, iar ciclarea se realizeaza cat timp contor<=*valf*, o valoare finala; la sfarsitul corpului ciclului, variabila *contor* este actualizata, fiind marita cu o valoare *pas*; pentru a nu se cicla la infinit, trebuie ca *pas*>0.	

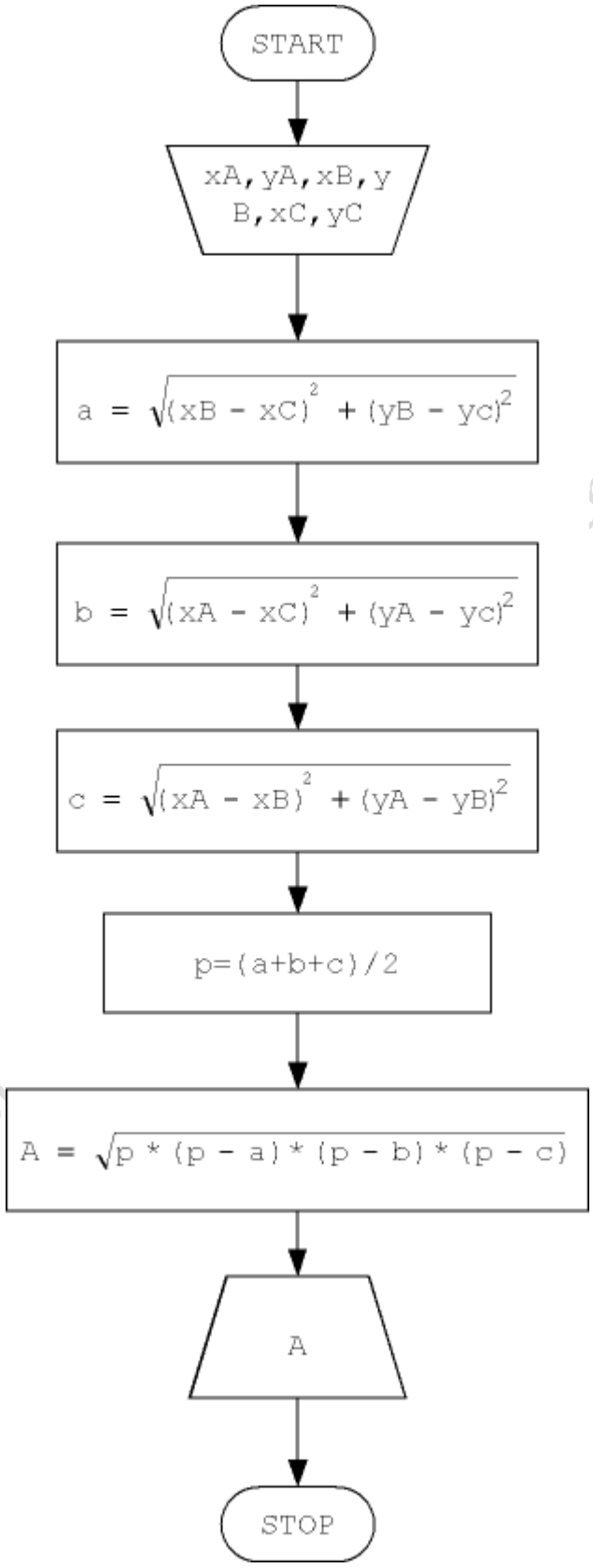
Tabelul 2.2 Structurile de control de baza si derivate

Scheme logice – Solutii pentru aplicatiile A2.1 – A2.20

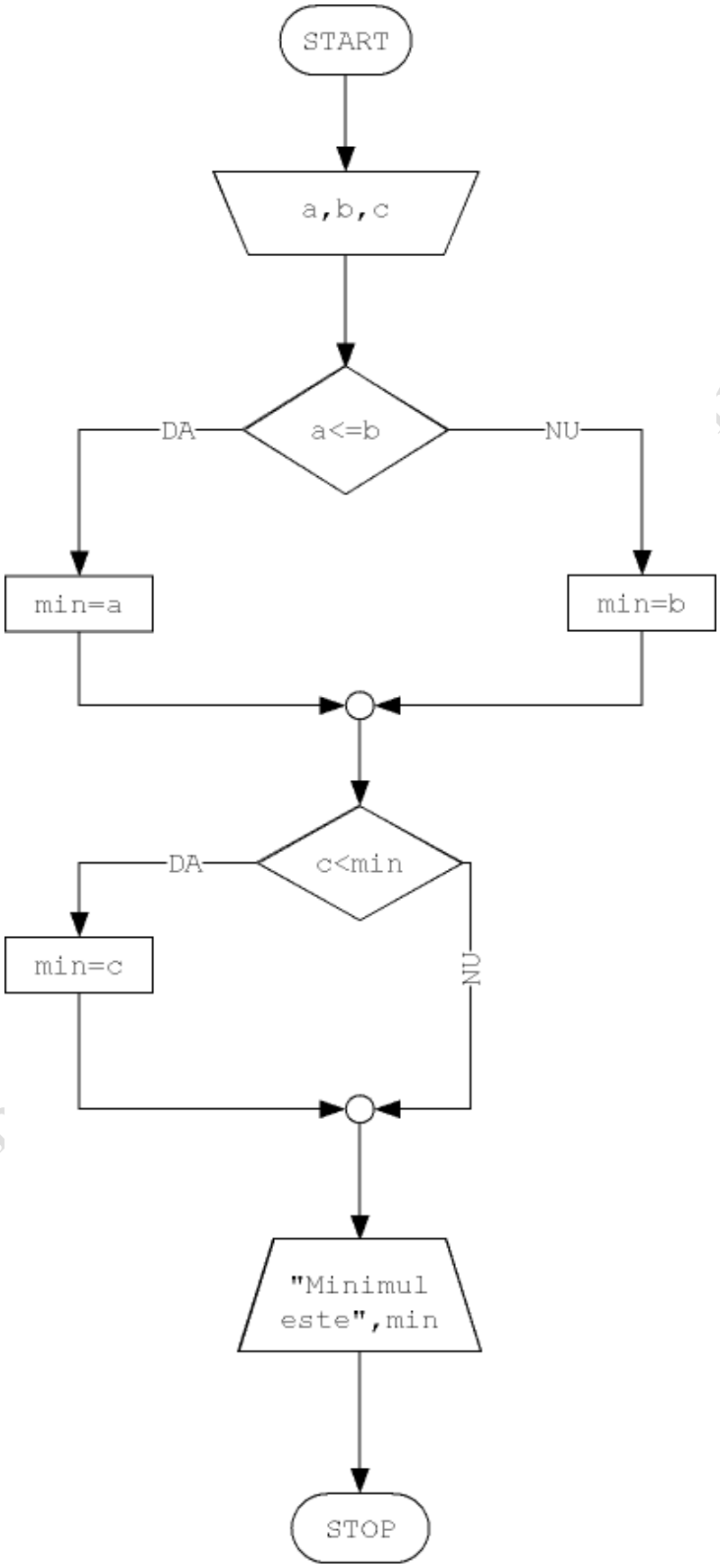
A2.1



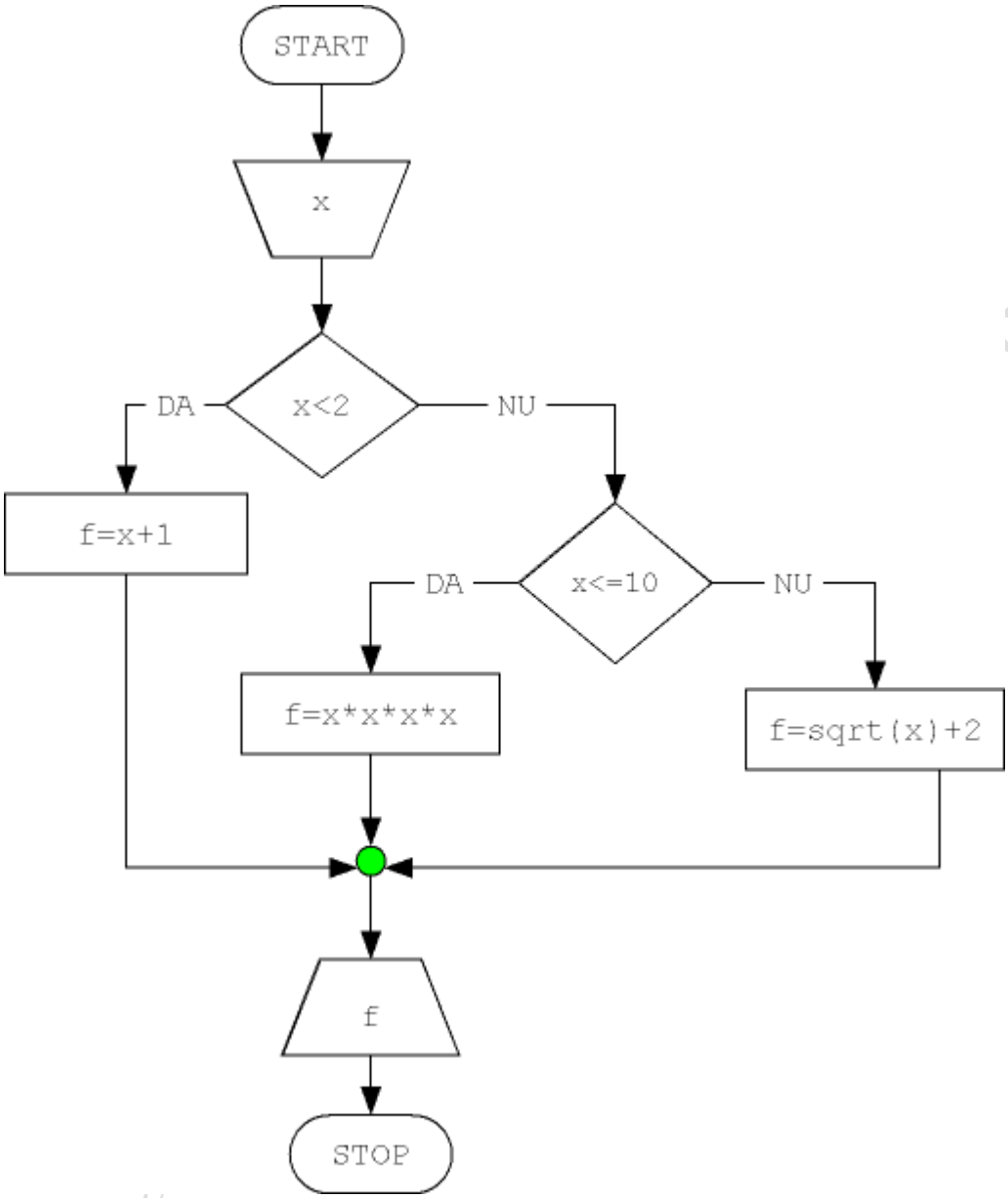
A2.2



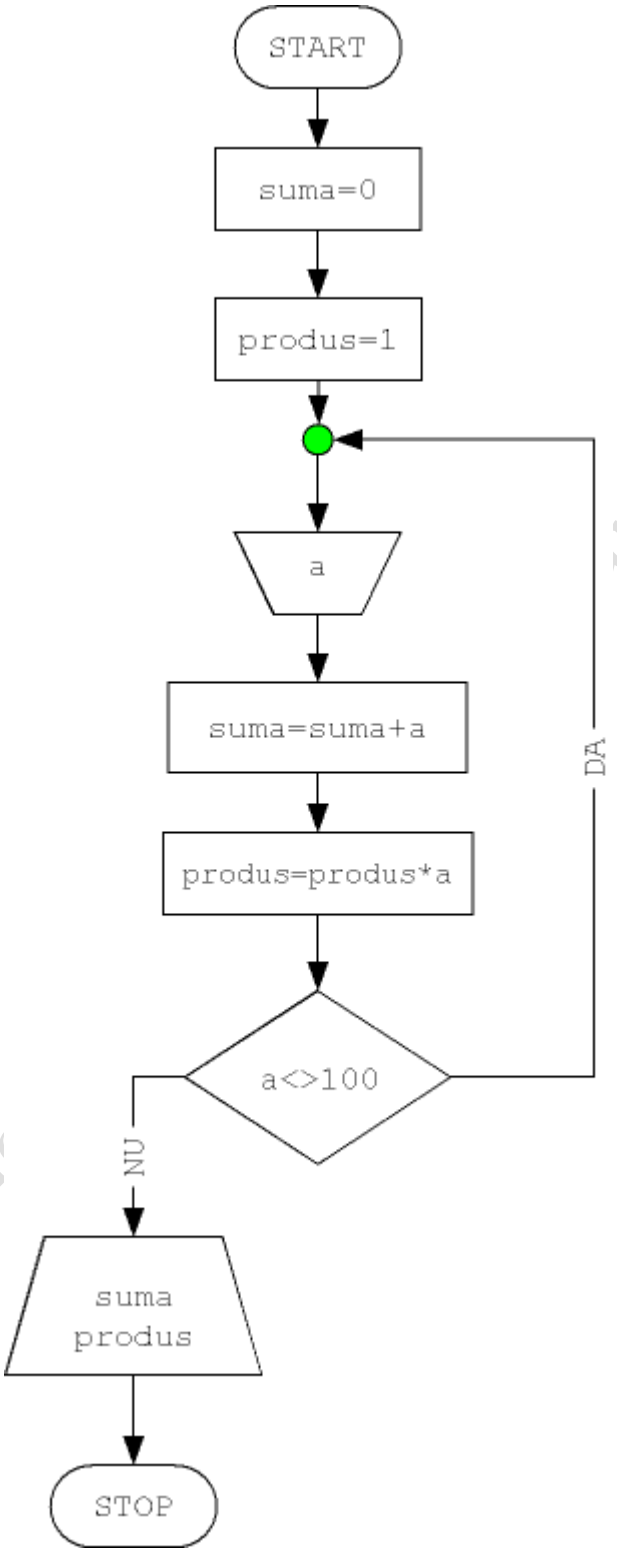
A2.3


START
a,b,c
a<=b
DA
NU
min=a
min=b
c<min
DA
NU
min=c
"Minimul este",min
STOP

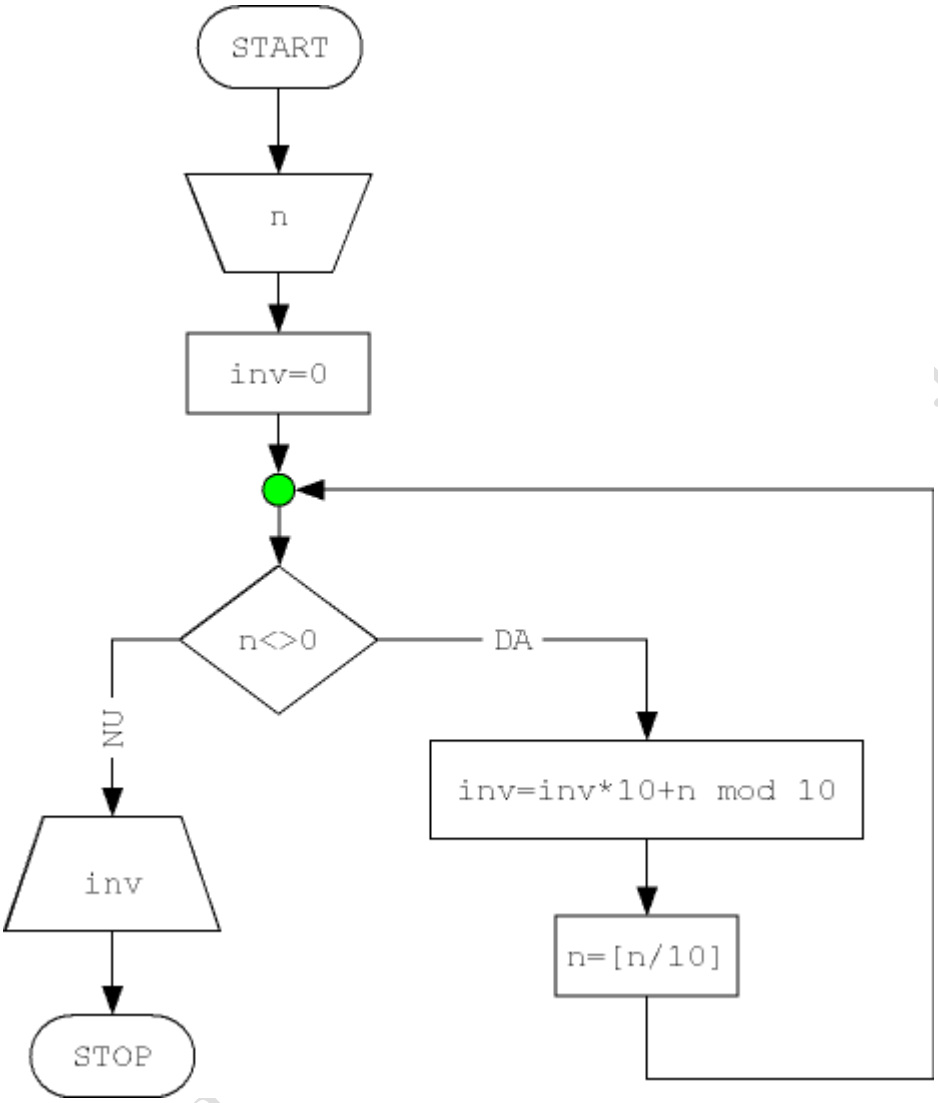
A2.4



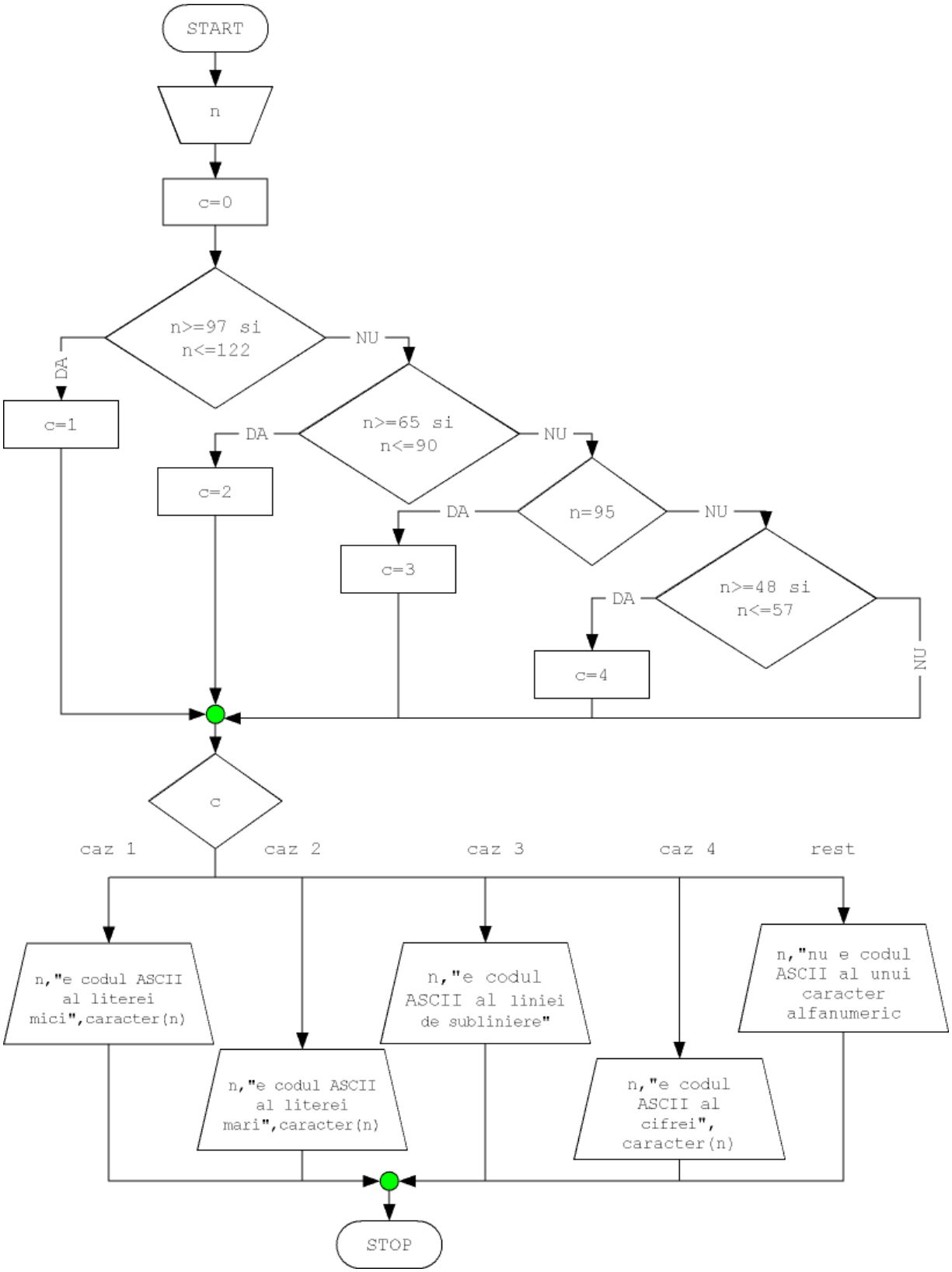
A2.5



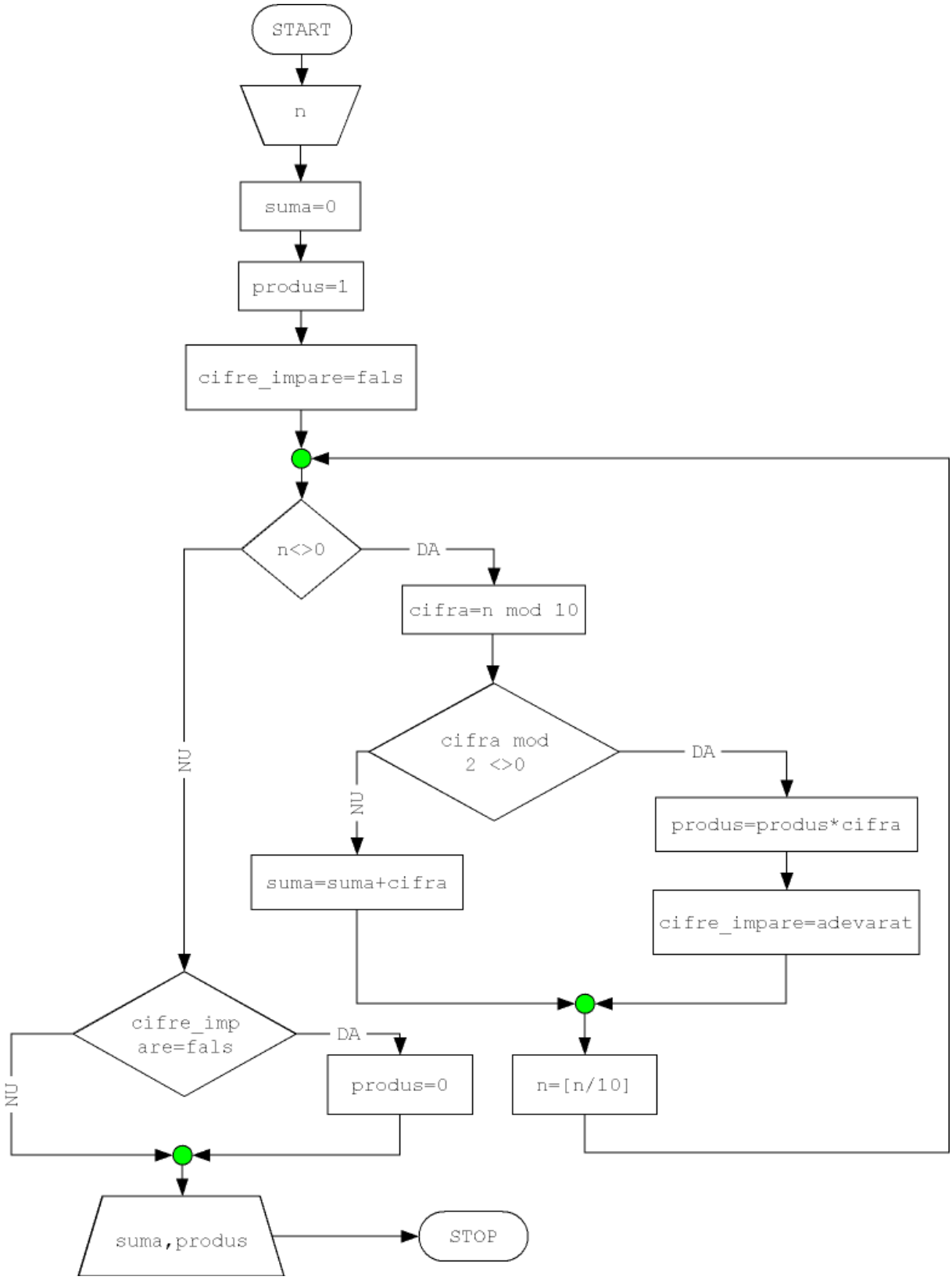
A2.6

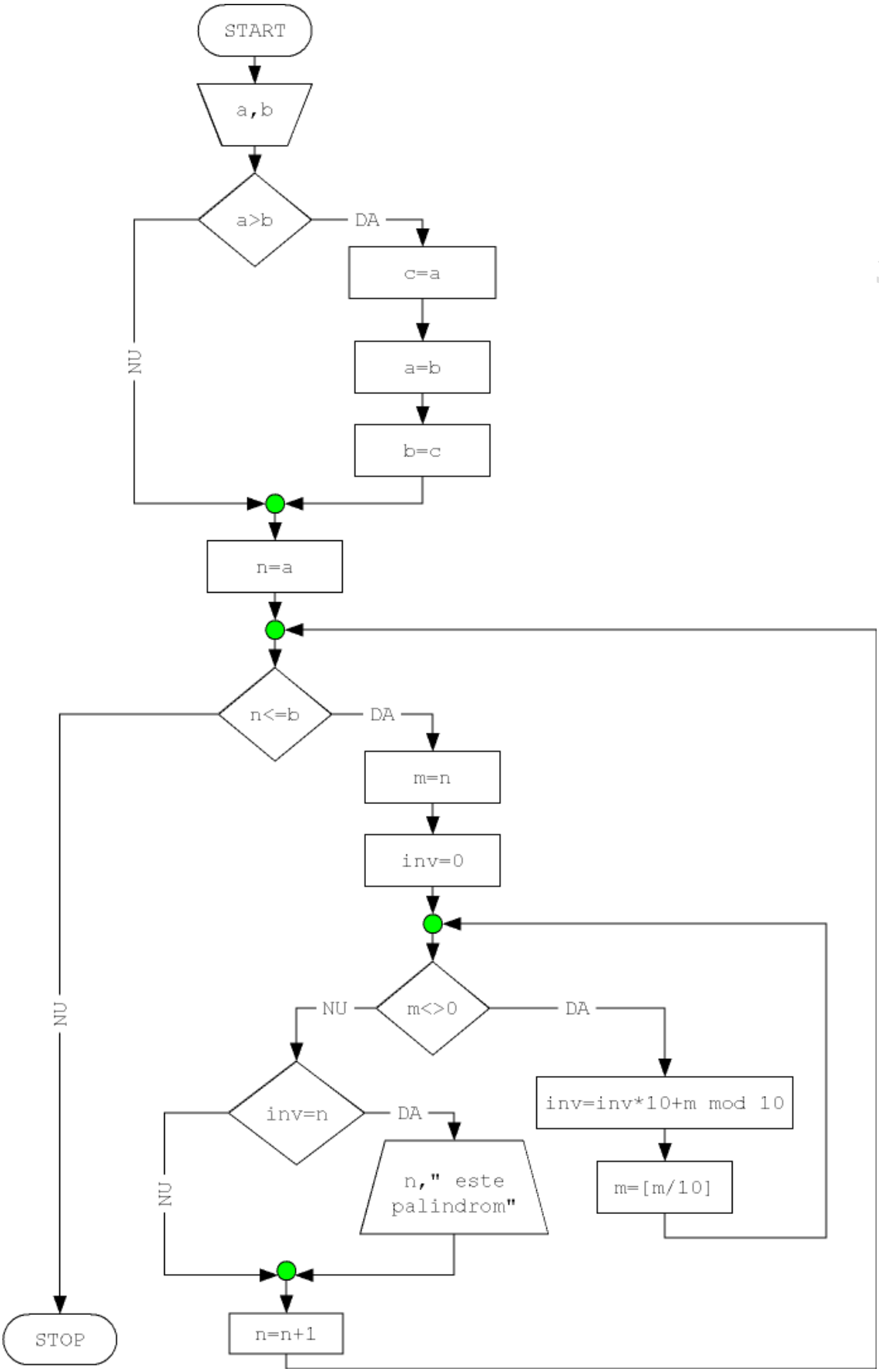


A2.7

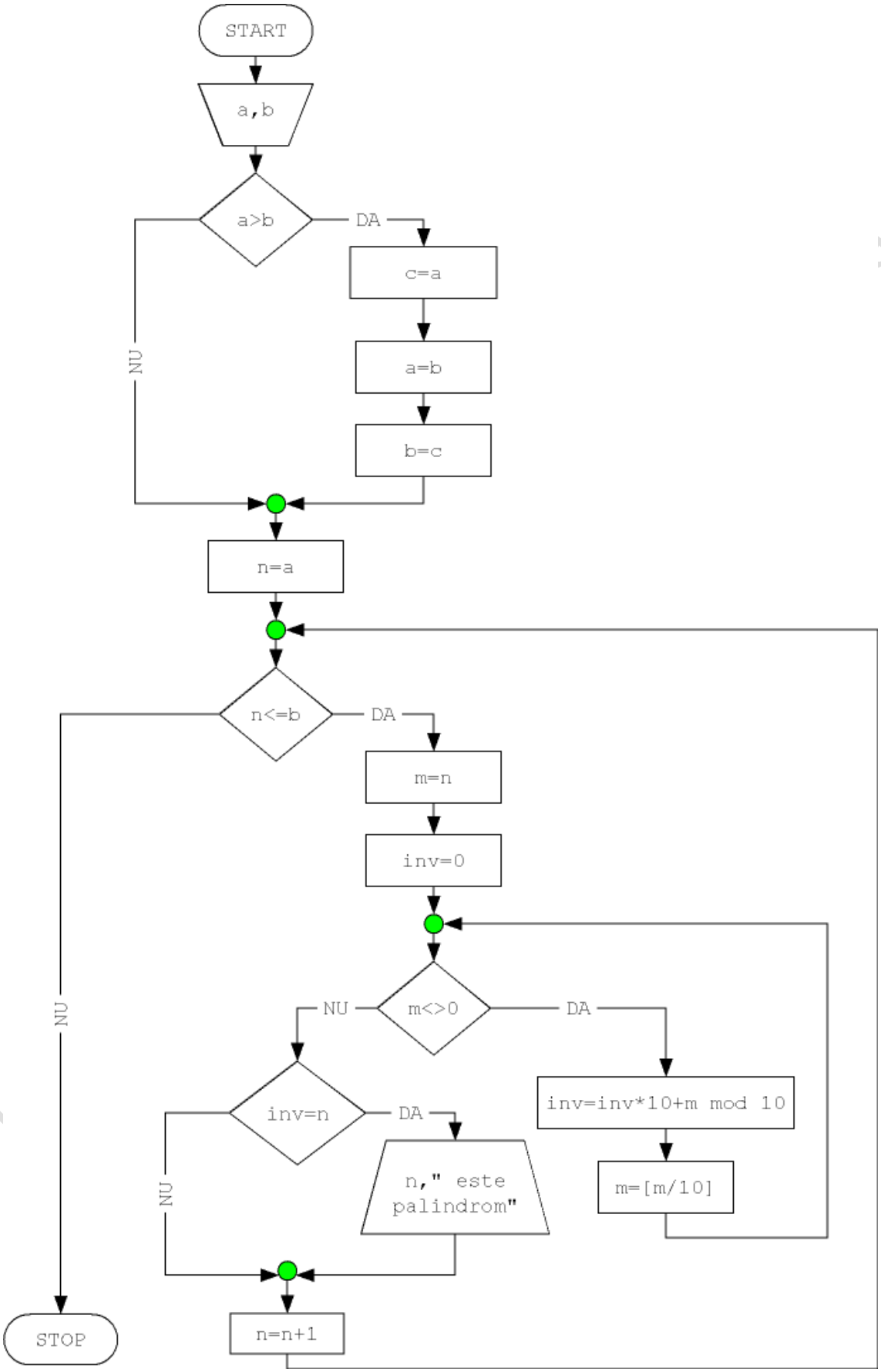

START
n
c=0
n>=97 si n<=122
DA
NU
c=1
n>=65 si n<=90
DA
NU
c=2
n=95
DA
NU
c=3
n>=48 si n<=57
DA
NU
c=4
c
caz 1
caz 2
caz 3
caz 4
rest
n,"e codul ASCII al literei mici",caracter(n)
n,"e codul ASCII al literei mari",caracter(n)
n,"e codul ASCII al liniei de subliniere"
n,"e codul ASCII al cifrei", caracter(n)
n,"nu e codul ASCII al unui caracter alfanumeric
STOP

A2.8

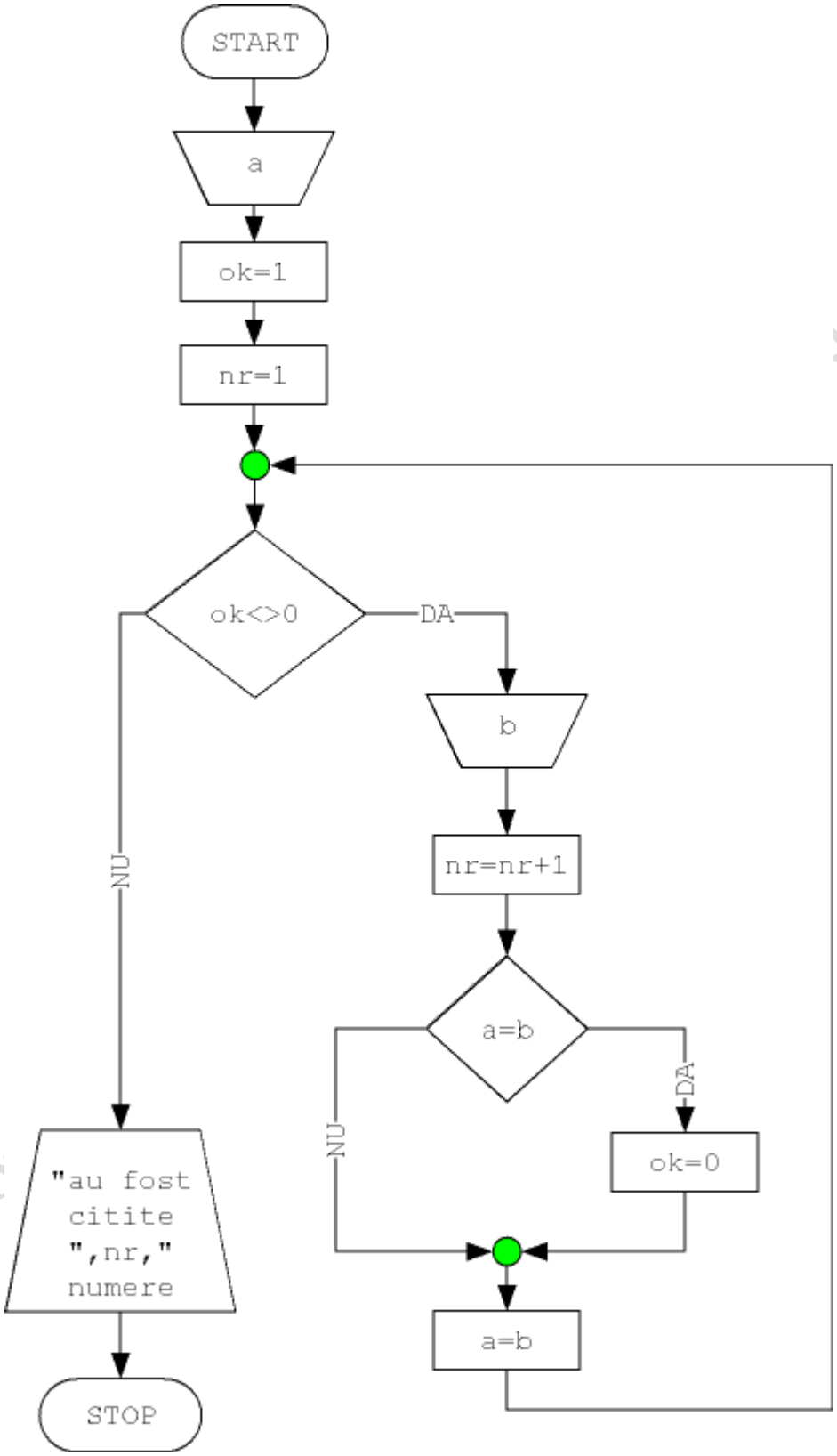

START
n
suma=0
produs=1
cifre_impare=fals
n<>0
DA
NU
cifra=n mod 10
cifra mod 2 <>0
DA
NU
produs=produs*cifra
suma=suma+cifra
cifre_impare=adevarat
cifre_imp are=fals
DA
NU
produs=0
n=[n/10]
suma,produs
STOP


START
a,b
a>b
DA
NU
c=a
a=b
b=c
n=a
n<=b
DA
NU
m=n
inv=0
m<>0
NU
DA
inv=n
DA
NU
inv=inv*10+m mod 10
n," este palindrom"
m=[m/10]
n=n+1
STOP

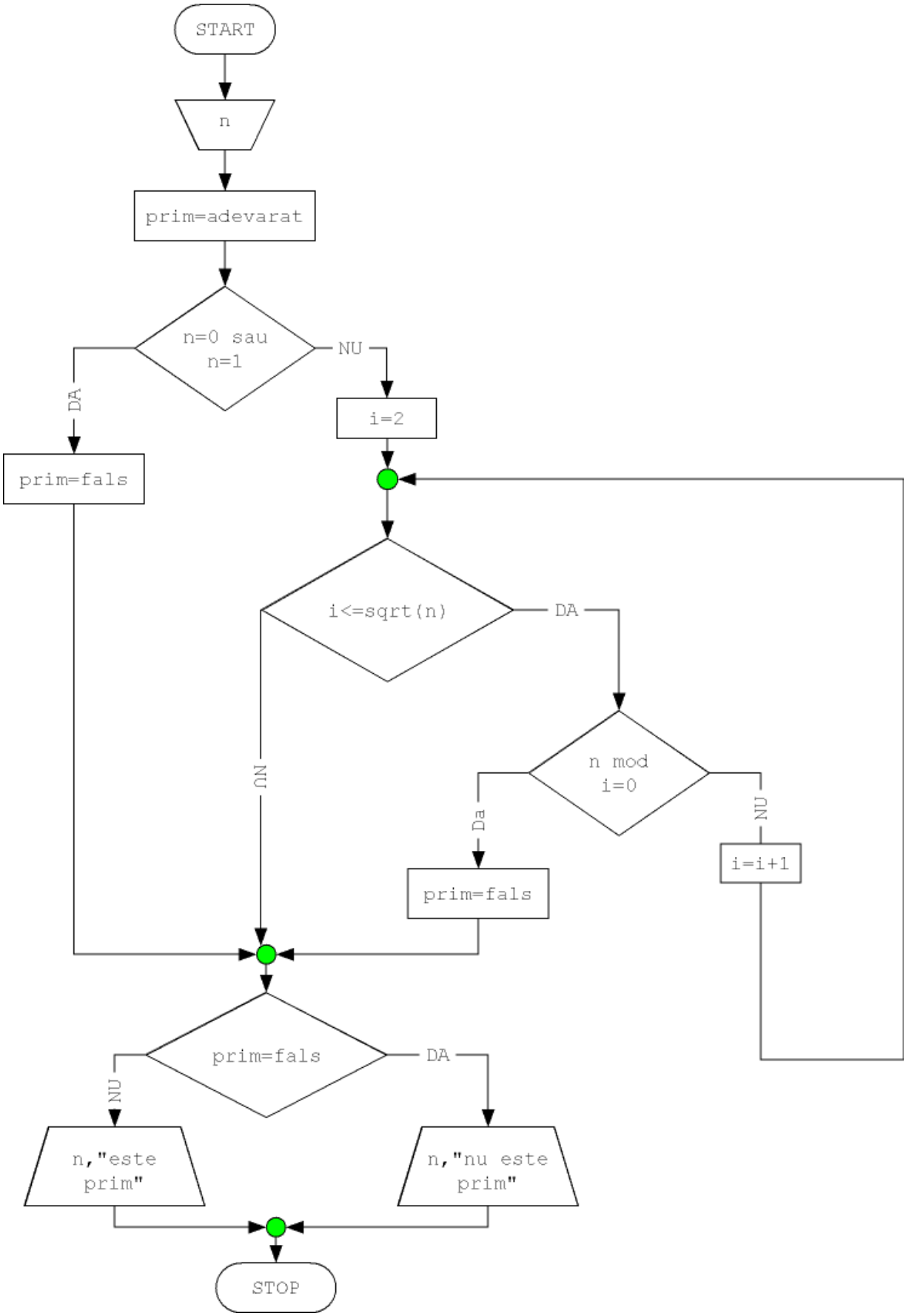
A2.9


START
a,b
a>b
DA
NU
c=a
a=b
b=c
n=a
n<=b
DA
NU
m=n
inv=0
m<>0
NU
DA
inv=n
DA
NU
inv=inv*10+m mod 10
n," este palindrom"
m=[m/10]
n=n+1
STOP

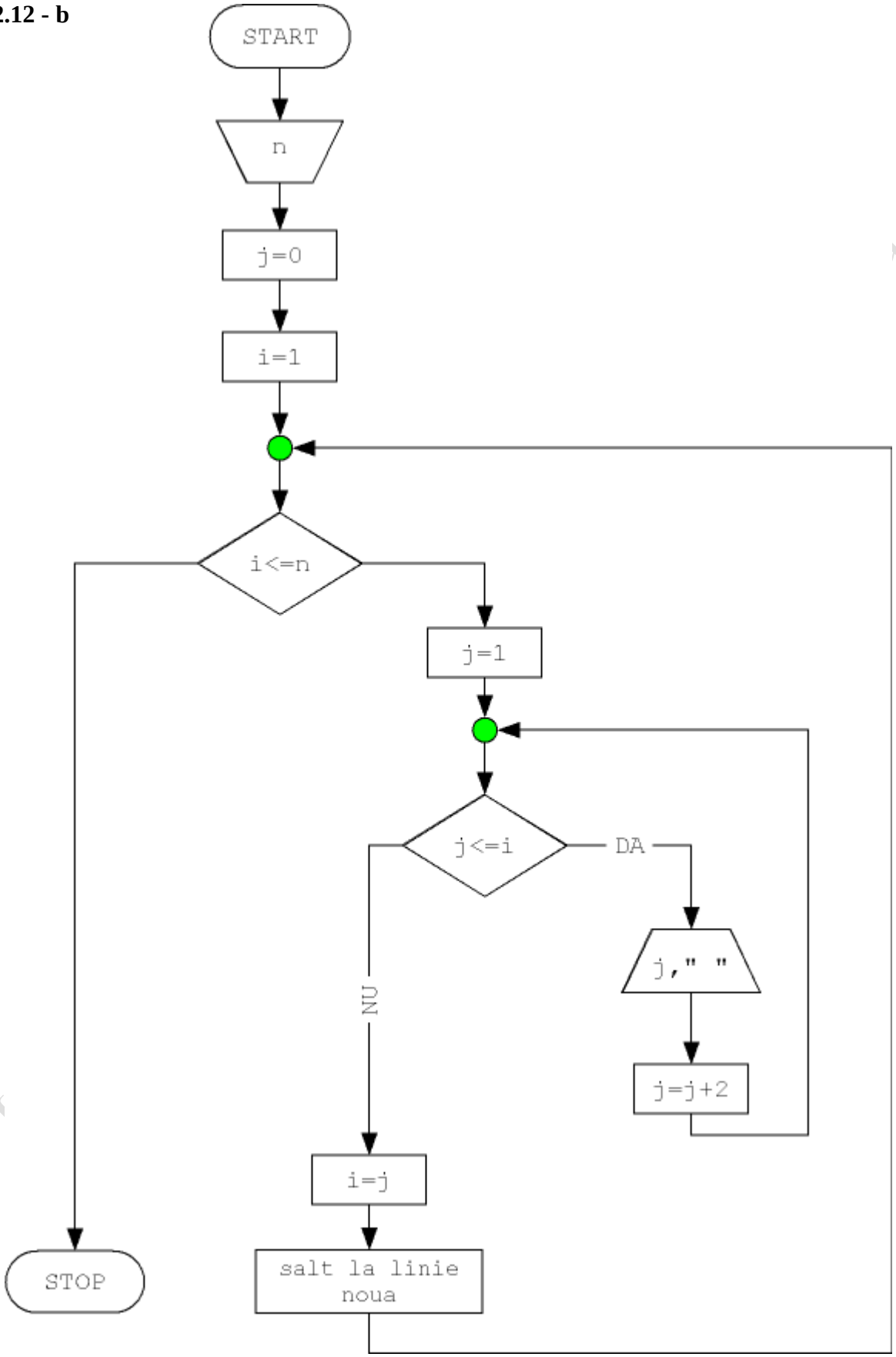
A2.10


START
a
ok=1
nr=1
ok<>0
DA
NU
b
nr=nr+1
a=b
DA
NU
ok=0
a=b
"au fost citite ",nr," numere
STOP

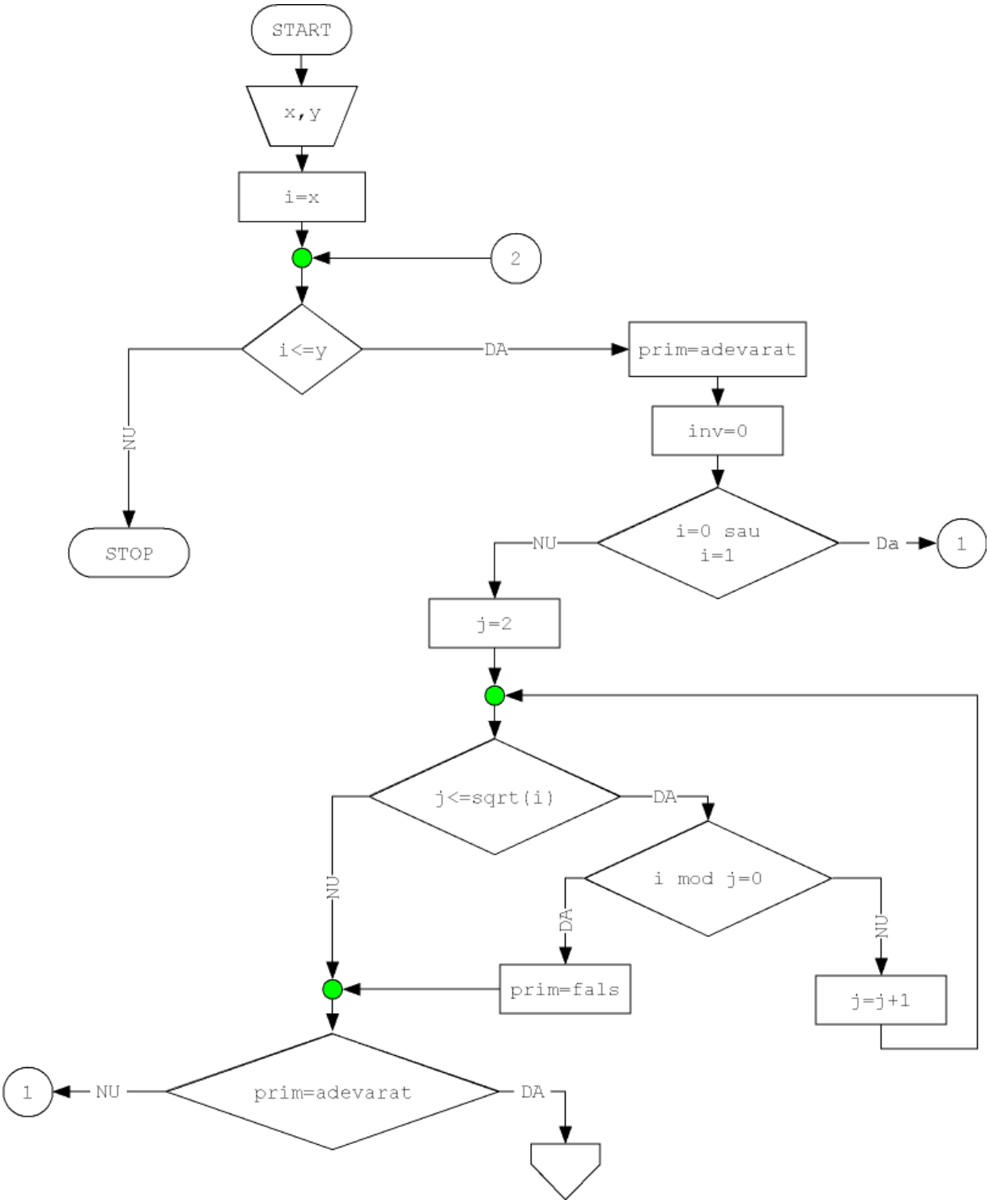
A2.11 ----------

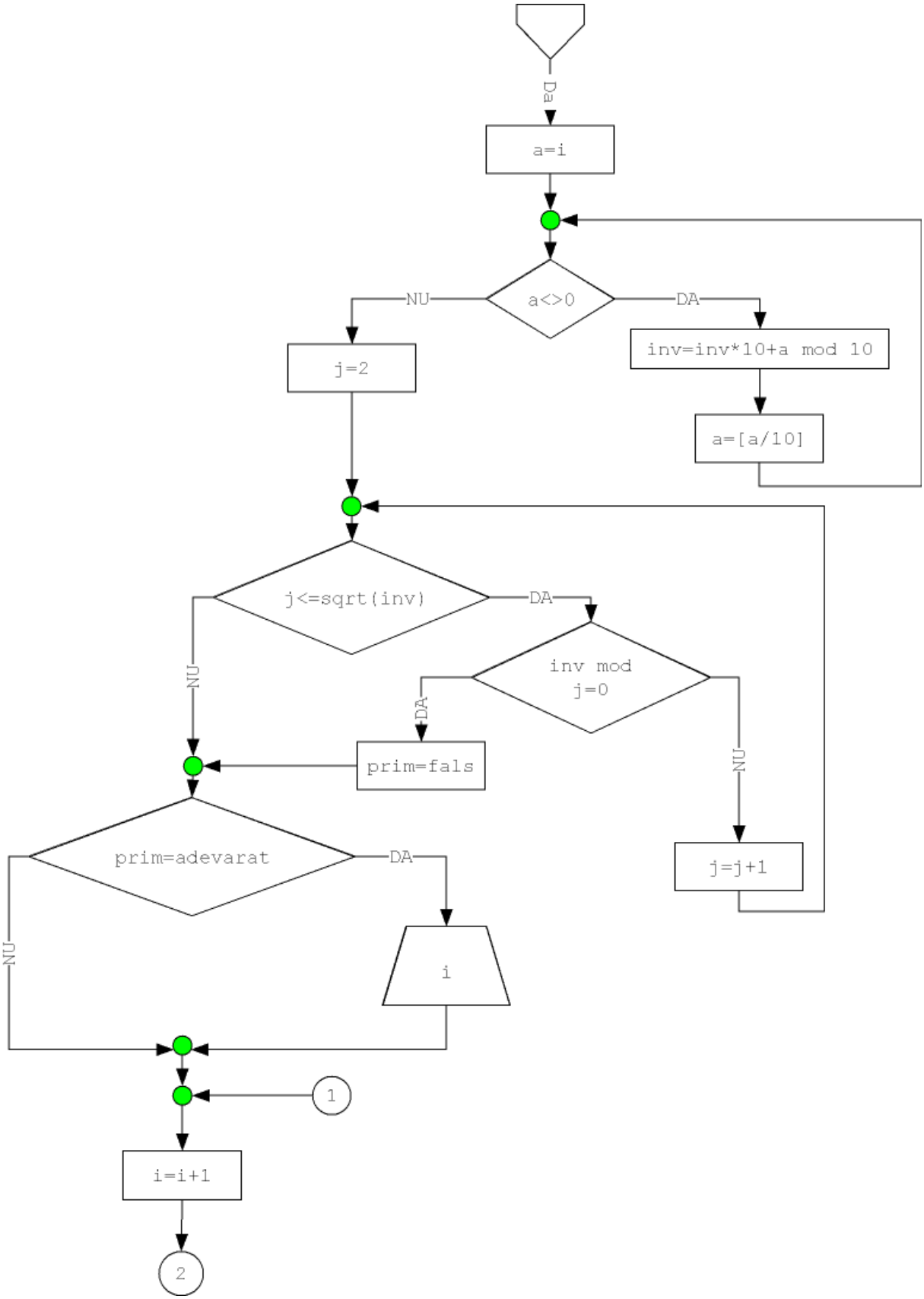

START
n
prim=adevarat
n=0 sau n=1
DA
NU
prim=fals
i=2
i<=sqrt(n)
DA
NU
n mod i=0
Da
NU
prim=fals
i=i+1
prim=fals
NU
DA
n,"este prim"
n,"nu este prim"
STOP

A2.12 - b

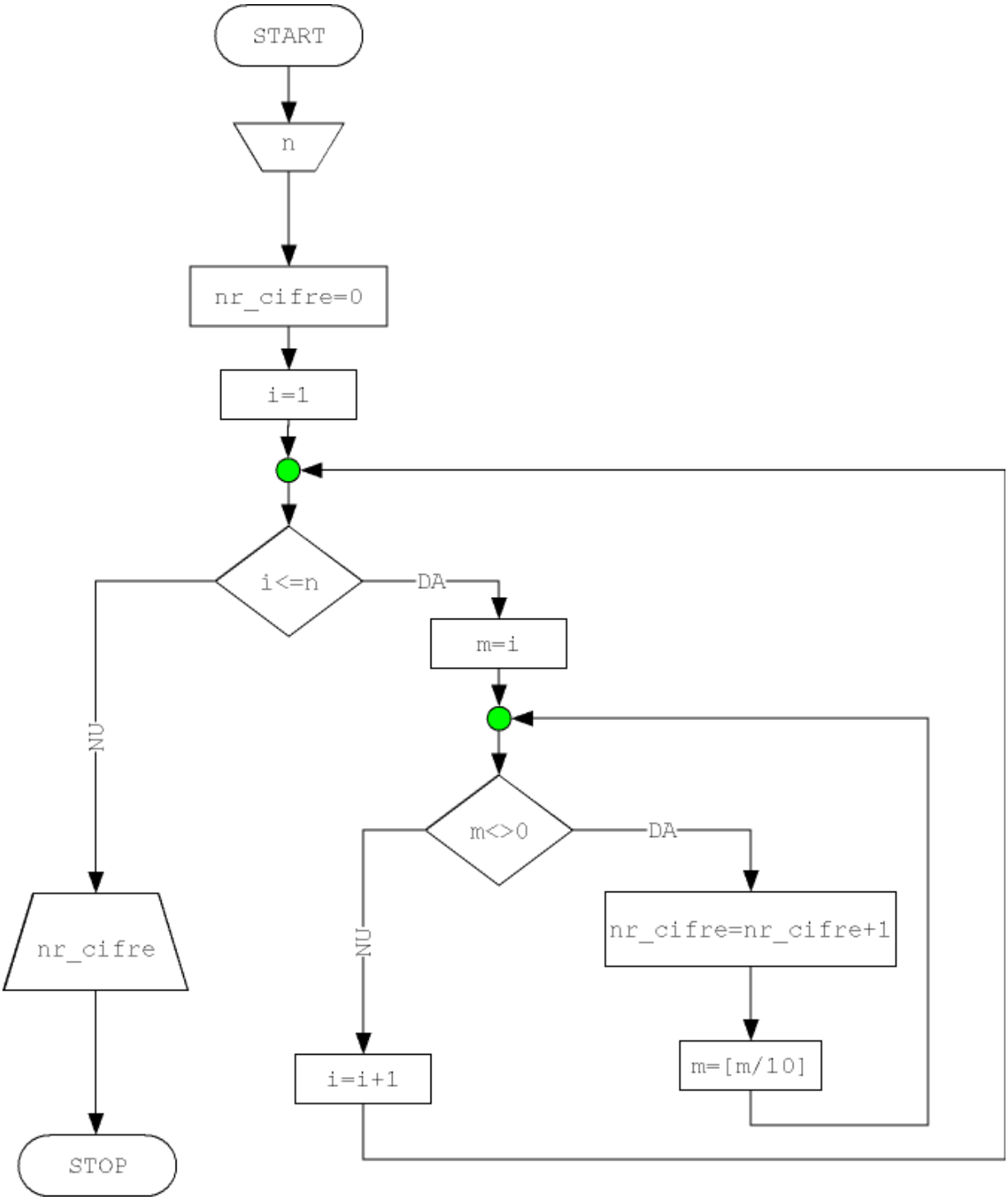


A2.13

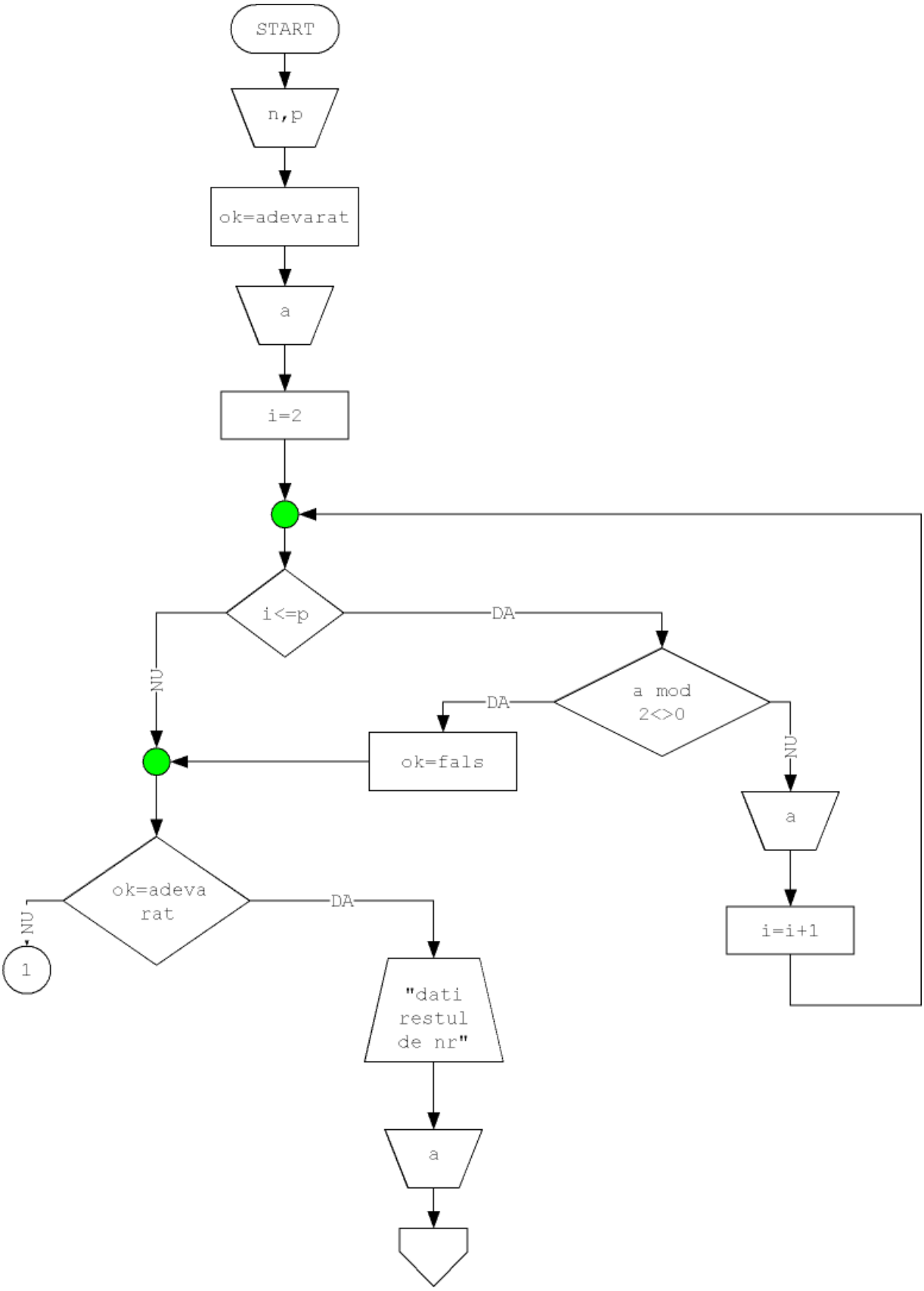

START
x,y
i=x
2
i<=y
DA
NU
prim=adevarat
inv=0
i=0 sau
i=1
NU
Da
1
STOP
j=2
j<=sqrt(i)
DA
NU
i mod j=0
DA
NU
prim=fals
j=j+1
1
NU
prim=adevarat
DA

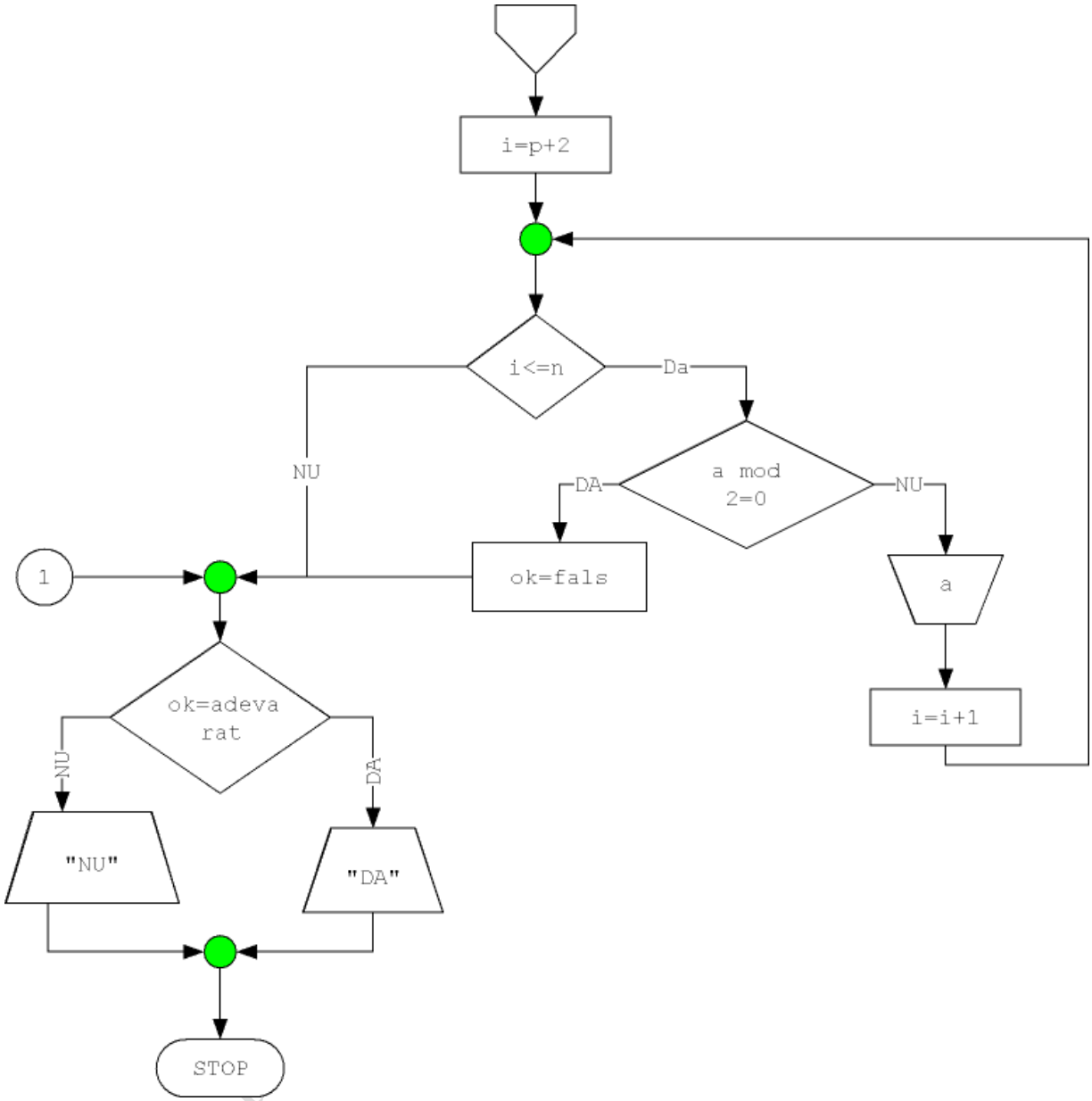

Da
a=i
a<>0
NU
DA
j=2
inv=inv*10+a mod 10
a=[a/10]
j<=sqrt(inv)
DA
NU
inv mod
j=0
DA
NU
prim=fals
prim=adevarat
DA
NU
j=j+1
i
1
i=i+1
2

A2.14

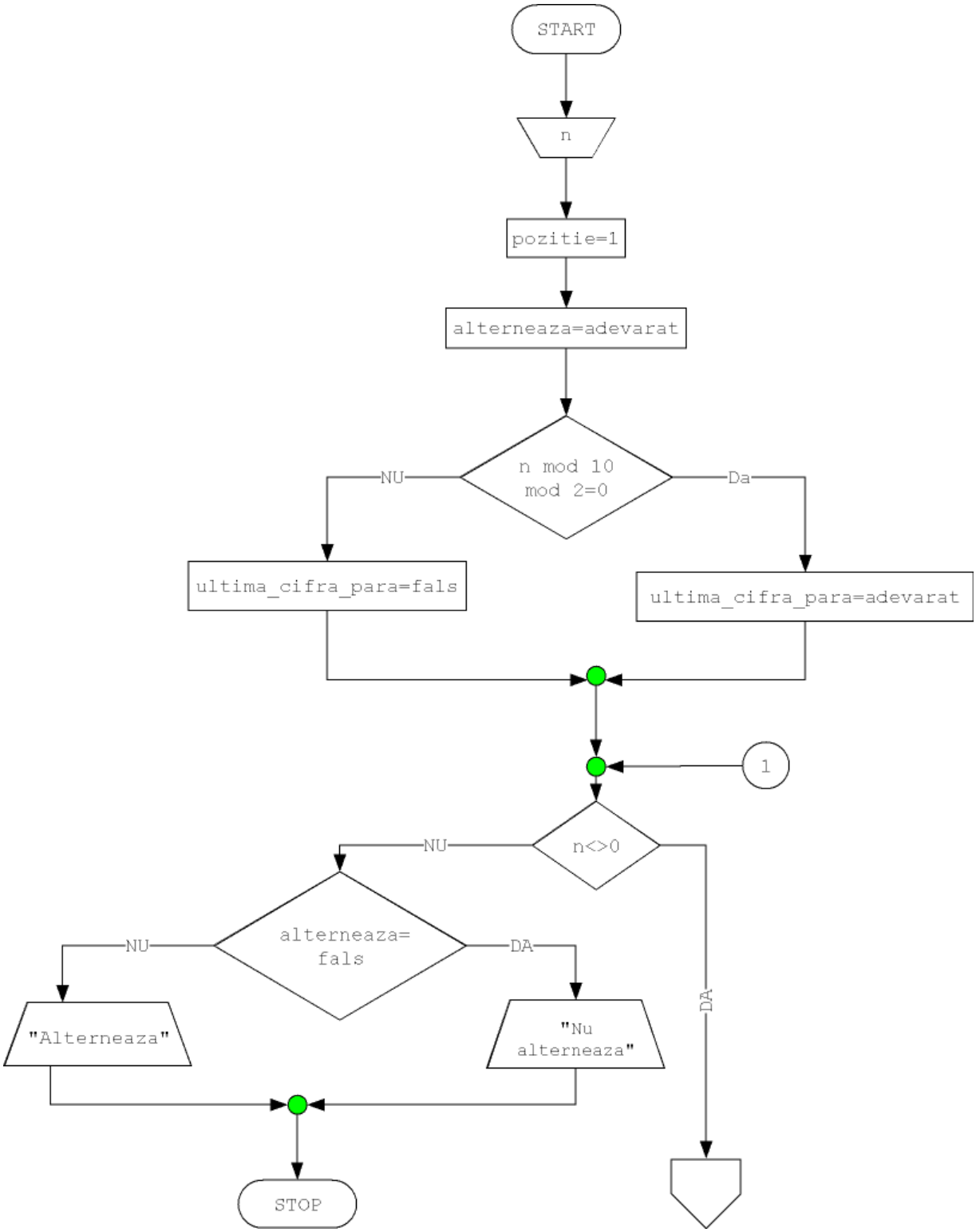

START
n
nr_cifre=0
i=1
i<=n
DA
NU
m=i
m<>0
DA
NU
nr_cifre=nr_cifre+1
m=[m/10]
i=i+1
nr_cifre
STOP

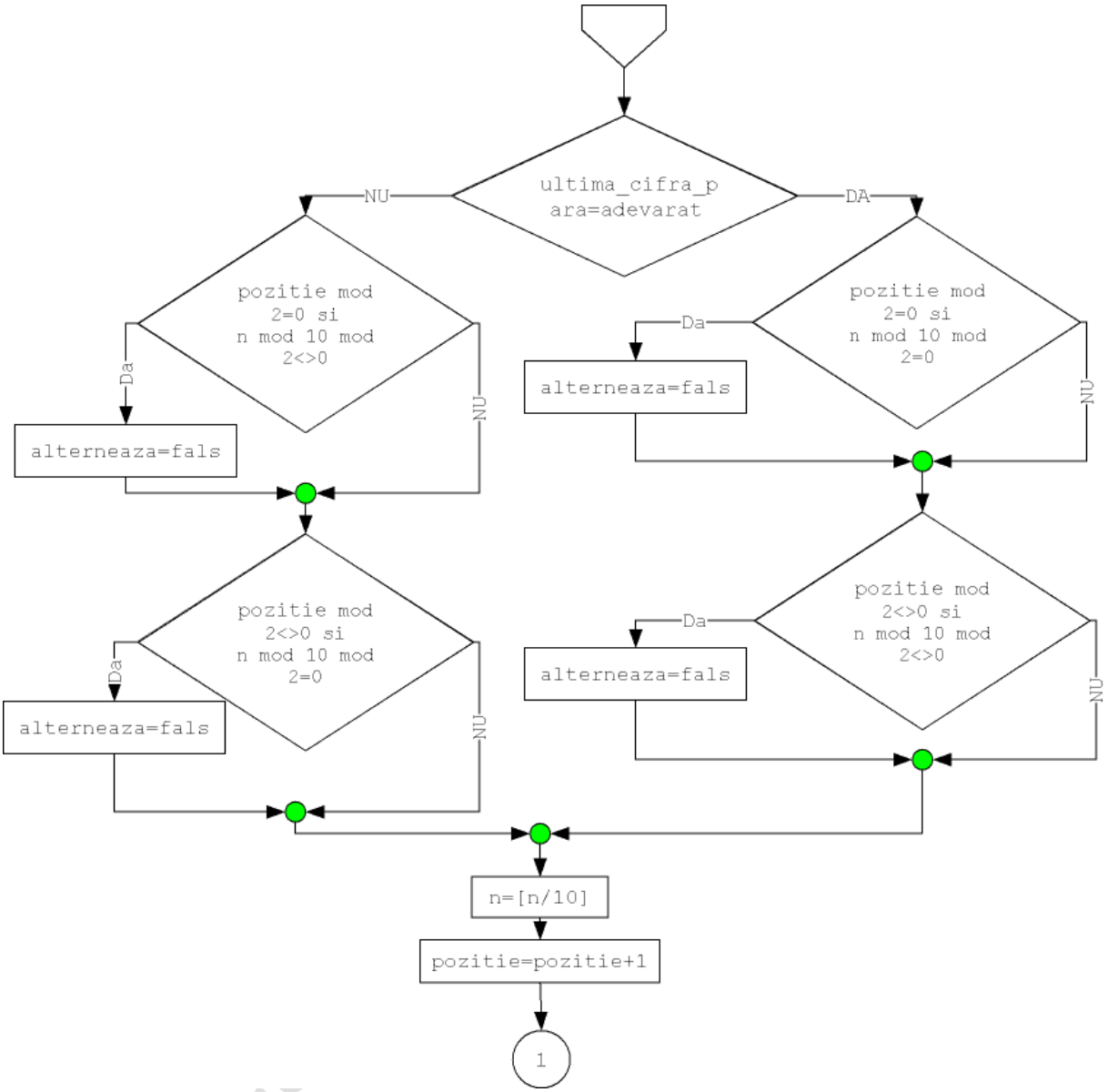
A2.15


START
n,p
ok=adevarat
a
i=2
i<=p
DA
NU
a mod 2<>0
DA
ok=fals
NU
a
i=i+1
ok=adeva rat
DA
NU
1
"dati restul de nr"
a

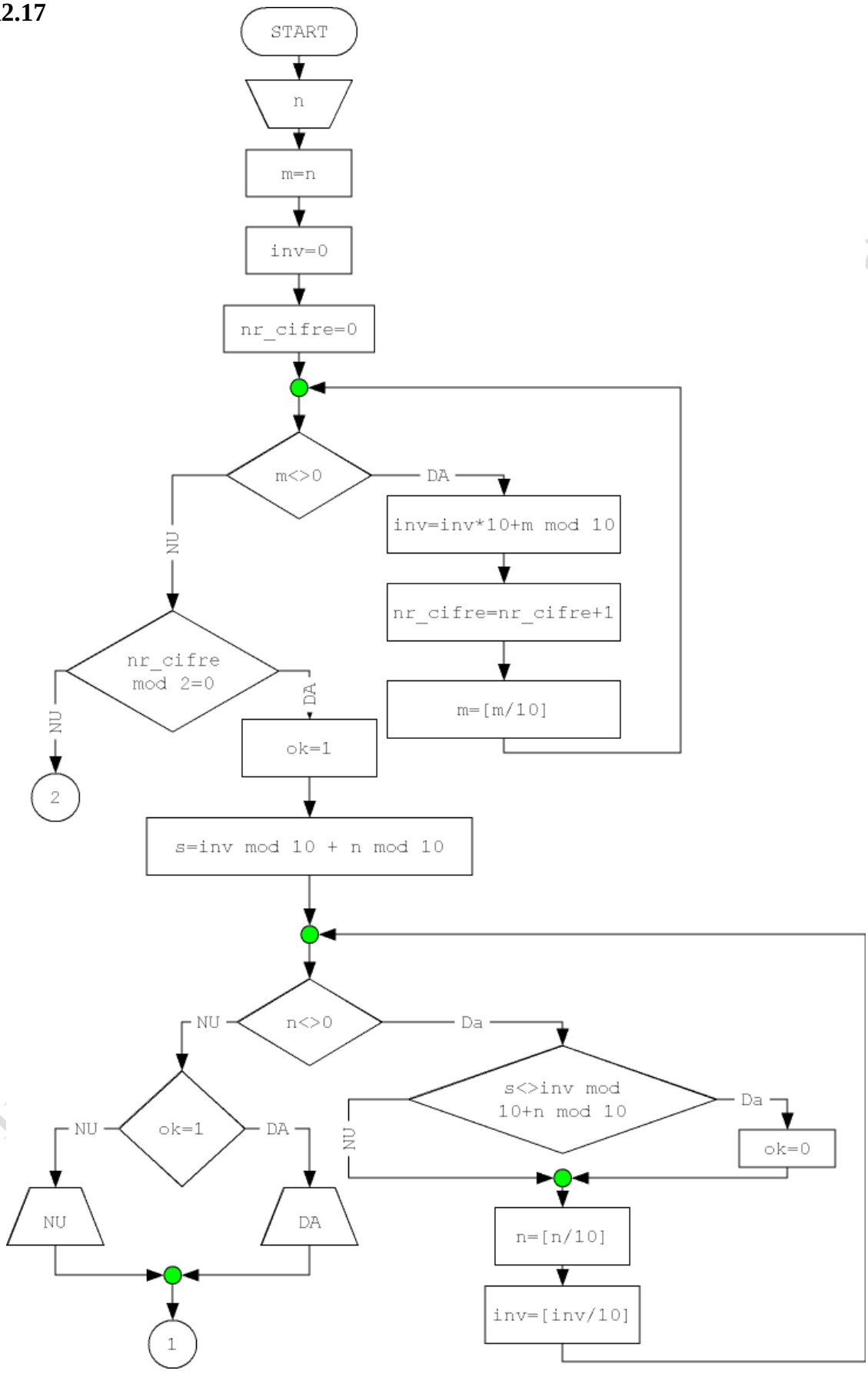

i=p+2
i<=n
Da
NU
a mod
2=0
DA
NU
1
ok=fals
a
i=i+1
ok=adeva
rat
NU
DA
"NU"
"DA"
STOP

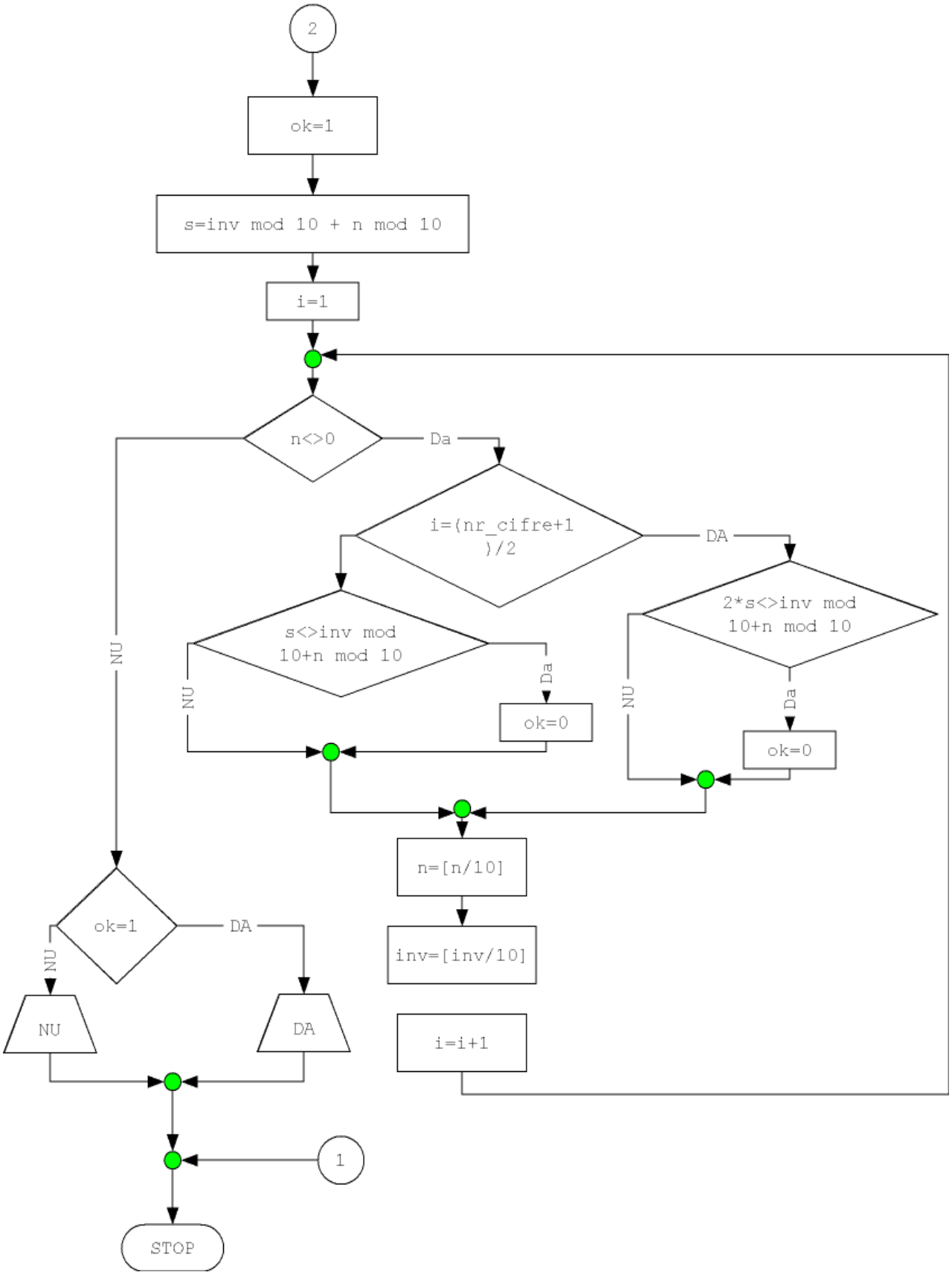
A2.16


START
n
pozitie=1
alterneaza=adevarat
n mod 10 mod 2=0
NU
Da
ultima_cifra_para=fals
ultima_cifra_para=adevarat
1
n<>0
NU
DA
alterneaza= fals
NU
DA
"Alterneaza"
"Nu alterneaza"
STOP

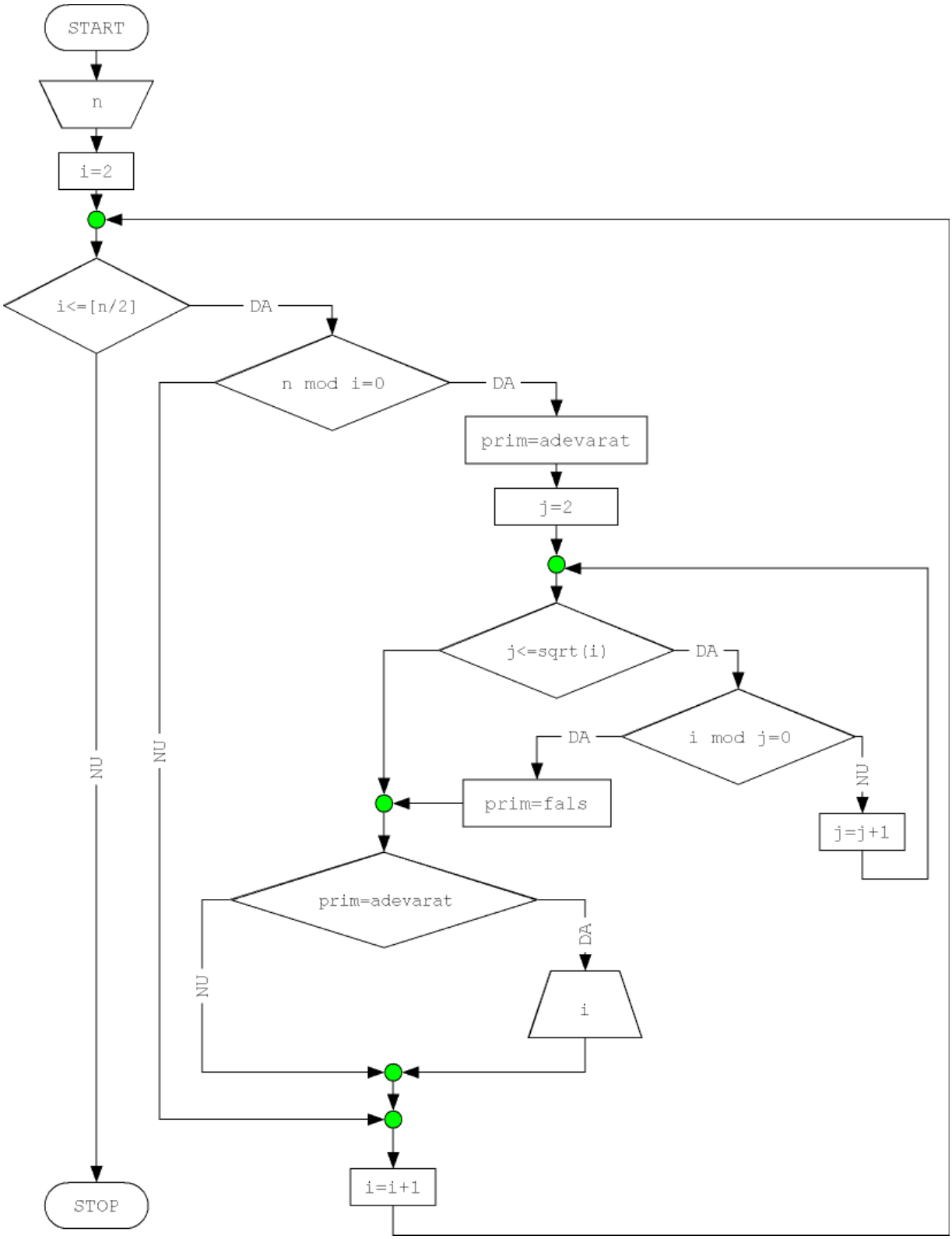

ultima_cifra_p
ara=adevarat
NU
DA
pozitie mod
2=0 si
n mod 10 mod
2<>0
Da
NU
alterneaza=fals
pozitie mod
2=0 si
n mod 10 mod
2=0
Da
NU
alterneaza=fals
pozitie mod
2<>0 si
n mod 10 mod
2=0
Da
NU
alterneaza=fals
pozitie mod
2<>0 si
n mod 10 mod
2<>0
Da
NU
alterneaza=fals
n=[n/10]
pozitie=pozitie+1
1

A2.17

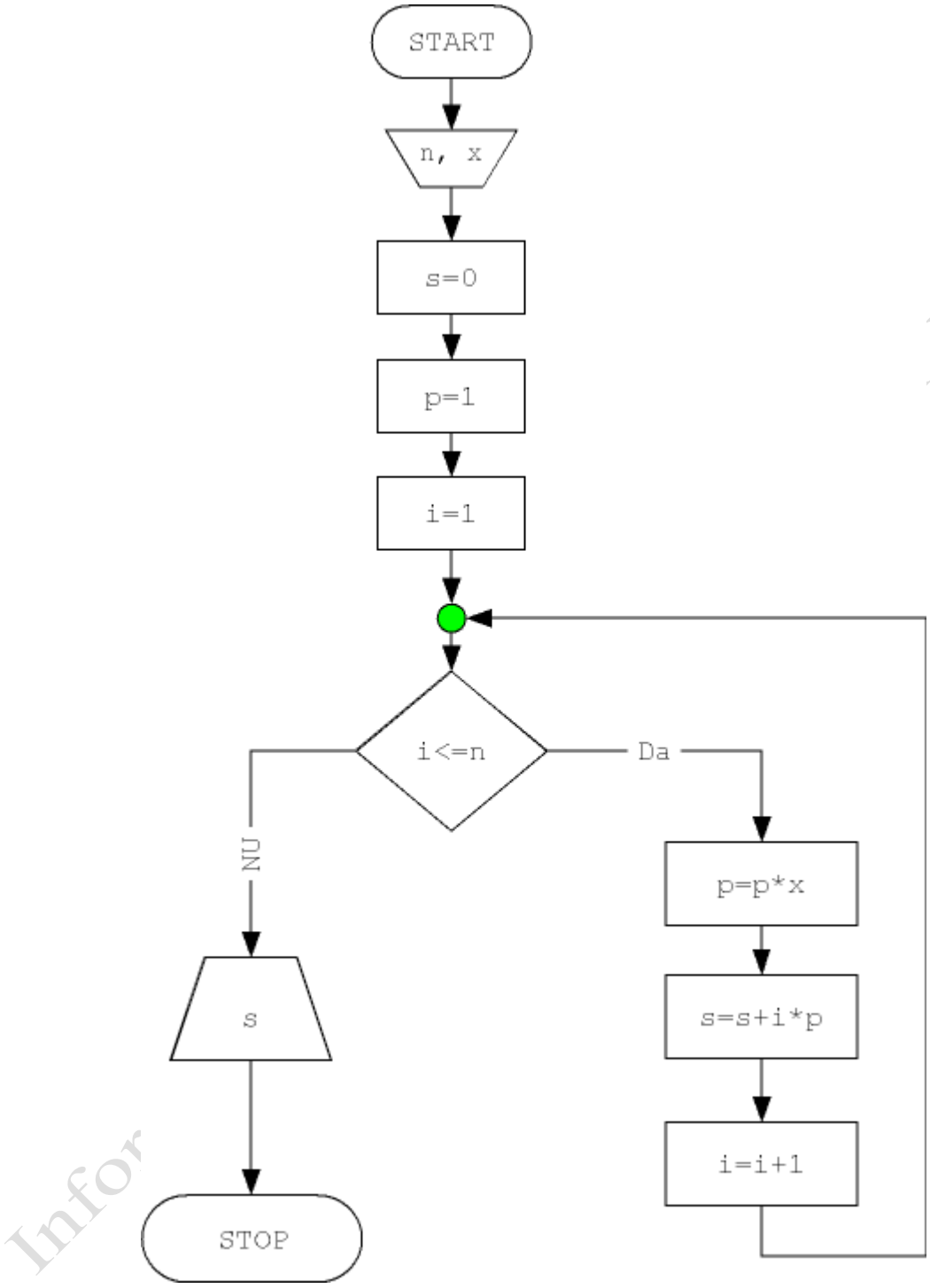

START
n
m=n
inv=0
nr_cifre=0
m<>0
DA
NU
inv=inv*10+m mod 10
nr_cifre=nr_cifre+1
m=[m/10]
nr_cifre
mod 2=0
DA
NU
ok=1
2
s=inv mod 10 + n mod 10
n<>0
NU
Da
ok=1
NU
DA
s<>inv mod
10+n mod 10
Da
NU
ok=0
NU
DA
n=[n/10]
inv=[inv/10]
1


2
ok=1
s=inv mod 10 + n mod 10
i=1
n<>0
Da
NU
i=(nr_cifre+1
)/2
DA
s<>inv mod
10+n mod 10
NU
Da
ok=0
2*s<>inv mod
10+n mod 10
NU
Da
ok=0
n=[n/10]
inv=[inv/10]
i=i+1
ok=1
NU
DA
NU
DA
1
STOP

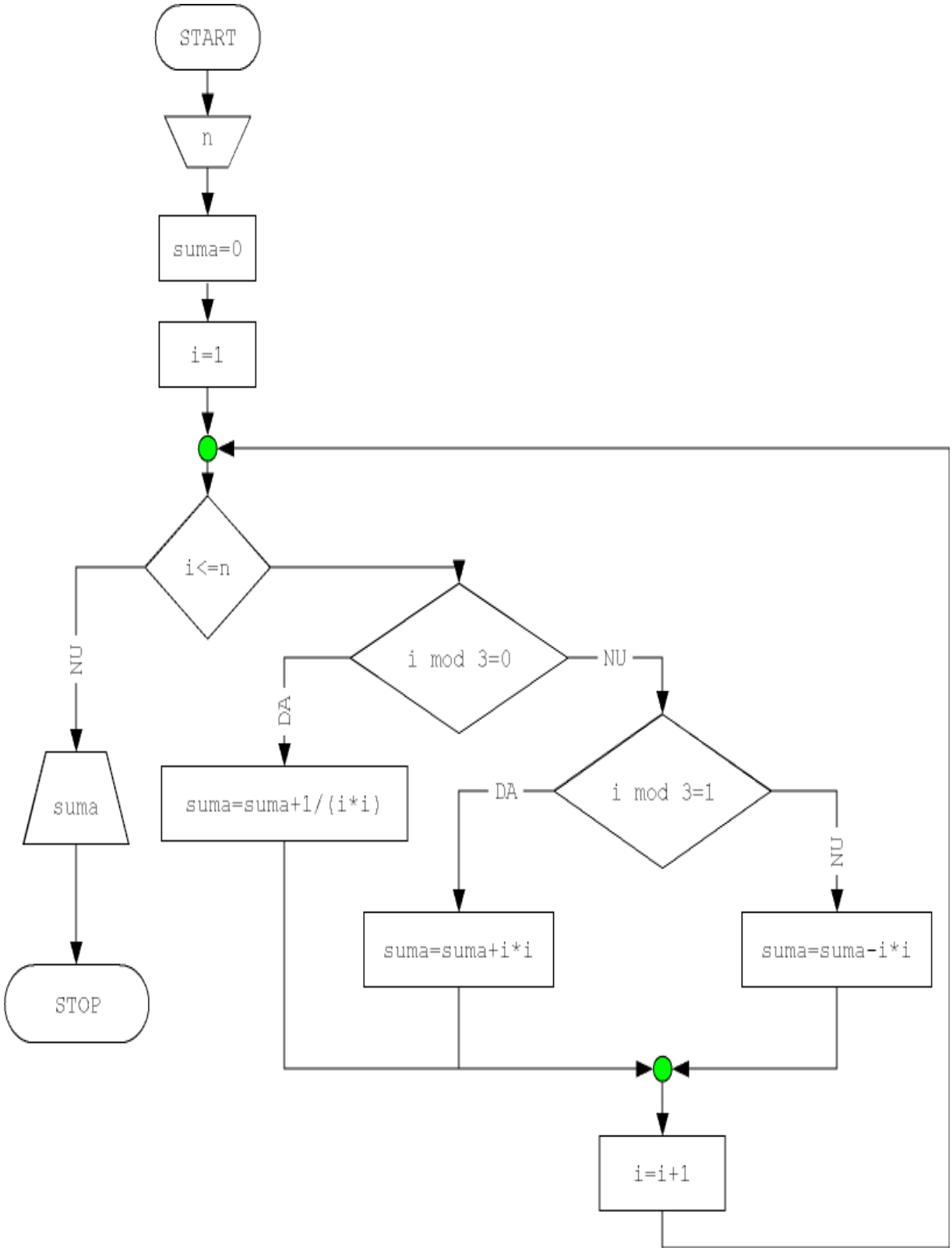
A2.18


START
n
i=2
i<=[n/2]
DA
NU
n mod i=0
DA
NU
prim=adevarat
j=2
j<=sqrt(i)
DA
i mod j=0
DA
NU
prim=fals
j=j+1
prim=adevarat
DA
NU
i
i=i+1
STOP

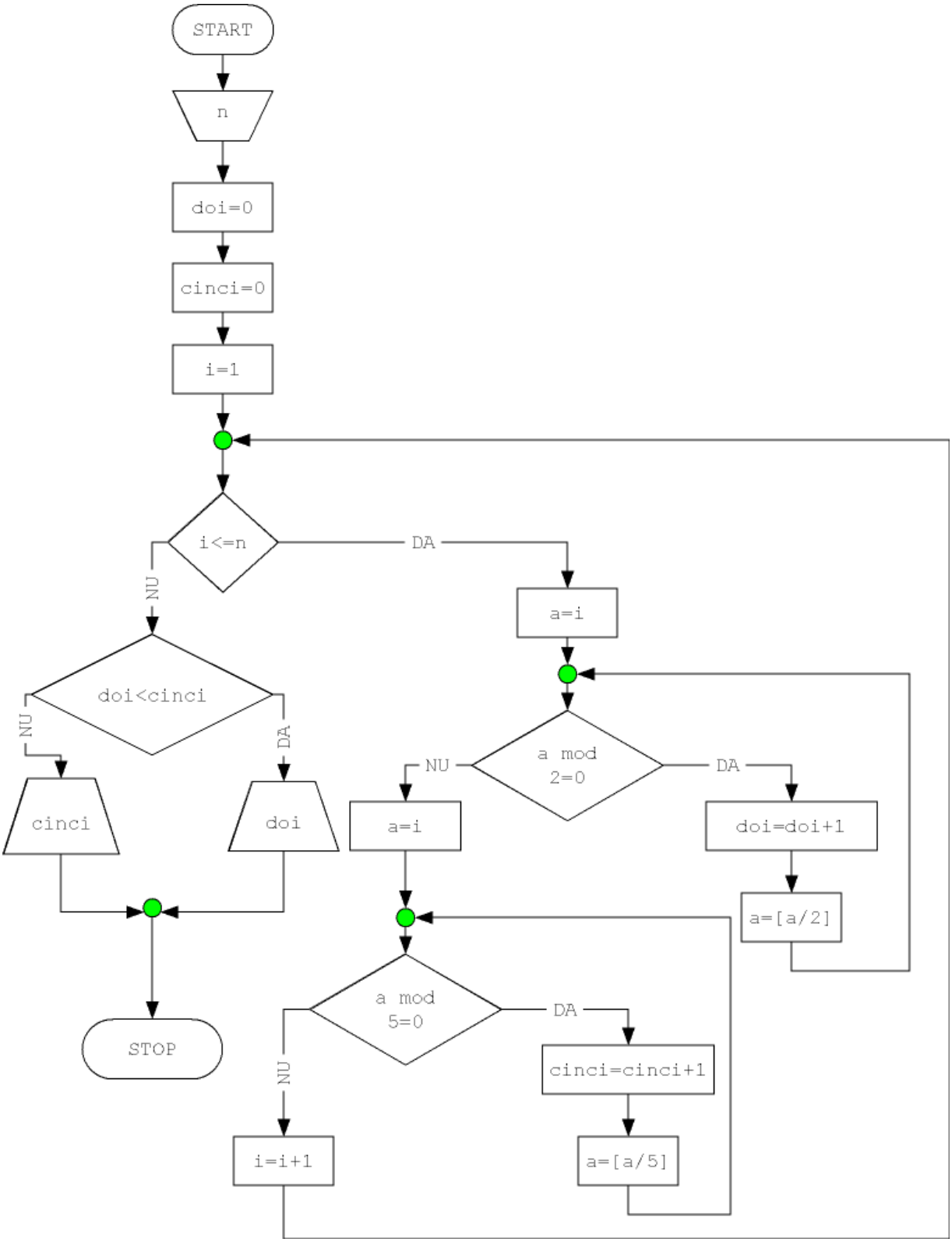
A2.19 – a



A2.19 – b


START
n
suma=0
i=1
i<=n
NU
suma
STOP
DA
i mod 3=0
NU
suma=suma+1/(i*i)
DA
i mod 3=1
NU
suma=suma+i*i
suma=suma-i*i
i=i+1

A2.20


START
n
doi=0
cinci=0
i=1
i<=n
DA
NU
a=i
doi<cinci
NU
DA
a mod 2=0
NU
DA
cinci
doi
a=i
doi=doi+1
a=[a/2]
a mod 5=0
DA
STOP
NU
cinci=cinci+1
i=i+1
a=[a/5]