



UNIUNEA EUROPEANĂ



GUVERNUL ROMÂNIEI



Instrumente Structurale
2007-2013



Platformă de e-learning și curriculum e-content pentru învățământul superior tehnic

Programare în limbaj de asamblare

51. Definirea și utilizarea de proceduri recursive.

Proceduri recursive (reentrante)

O procedură care se apelează pe ea însăși (direct sau indirect) se numește procedură recursivă (sau reentrantă). Pentru ca apelul să fie corect este necesar ca procedura:

să salveze în stivă indicatorii de condiții și registrele utilizate;

datele referite să fie în registre sau în stivă, dar nu în memorie;

să conțină o condiție de oprire a apelului recursiv.

Recursivitatea este utilizată când fie algoritmul, fie datele sunt descrise recursiv.

1) Procedura citește un șir de caractere (o linie de text) pe care, apoi, le tipărește în ordine inversă:

```
proc_scrie_inv      proc
    mov ah, 1      ; citește un caracter
    int 21h
    cmp al, 0dh    ; se compară cu sfârșit de linie
    jnz citeste    ; dacă nu e sfârșit linie citește caracter
    ret           ; dacă e sfârșit linie procedura s-a terminat
citeste: push ax   ; se depune caracterul în stivă
                call proc_scrie_inv ; se apelează recursiv
                pop ax           ; la revenire din apel se ia car. din stivă
                push dx          ; salvare registru folosit pentru tipărire
                mov dl, al       ; caracterul de tipărit se transmite în (DL)
                mov ah, 2       ; și se tipărește
                int 21h
                pop dx           ; refacerea registrului
                ret
proc_scrie_inv      endp
```

((SS):(SP))	→	caracter-ASCII	↑ Adrese mici
		IP_rev_procedură	
		...	
		caracter-ASCII	
		IP_rev_procedură	
		caracter-ASCII	
		IP_rev_prog_principal	↓ Adrese mari

2a) Procedură pentru calculul recursiv al factorialului.

```
factorial      proc
```

```
; procedura primește valoarea lui N în registrul AX
```

```
; rezultatul va rămâne în registrul AX
```

```
    cld           ; inițializare (CF=0), la intrarea în procedură, sau
                ; în prg. principal, înainte de apelul procedurii
    pushf        ; salvare indicatori
    push dx      ; și a registrului (DX), modificat de înmulțire
    push bp
    mov bp, sp   ; inițializare (BP), pentru a lua, la revenire, K-ul
    cmp ax, 1    ; test de sfârșit apel recursiv
```

```

je gata_apel
push ax      ; se depune termenul curent, K, în stivă
dec ax      ; termenul următor din factorial, K-1
call factorial ; apelul recursiv
jc poz_cf   ; depășirea de la apelul anterior se propagă
mul word ptr [bp-2] ; (AX) * (K-vârf_stivă) = (DX,AX)
jnc cont    ; dacă nu a apărut depășire, se continuă
poz_cf:
or word ptr [bp+4], 1 ; poz. (CF=1, imaginea din stivă)
cont:
add sp, 2 ; descărcarea lui K, din vârful stivei
gata_apel:
pop bp    ; refacerea registrelor salvate
pop dx
popf     ; refacerea indicatorilor
ret      ; revenire din apel
factorial endp

```

Imaginea stivei:

```

bp  →  bp_vechi
      dx_vechi
      indicatori
      adr_rev_proc
      N - 1
bp  →  ...
      N
bp  →  bp_vechi
      dx_vechi
      indicatori
      adr_rev_prog_principal

```

2b) Procedură recursivă de calcul al factorialului, dar valoarea lui N se transmite prin stivă și se returnează în registrul (AX).

```

fact proc
clc ; inițializare (CF=0), la intrarea în procedură sau
    ; în prog. principal, înainte de apelul procedurii
pushf ; salvare indicatori și registre afectate
push dx
push bp
mov bp, sp ; inițializare (BP), pentru a lua, la revenire, K-ul
mov ax, [bp+8] ; citește valoarea lui N (K), din stivă
cmp ax, 1 ; test sfârșit apel recursiv
je gata_apel
dec ax ; se trece la elementul următor, N-1 (K-1)
push ax ; se depune acest termen în stivă
call fact ; apelul recursiv

```

```

    jc poz_cf
    mul word ptr [bp+8] ; rez. înmulțirii este în (DX,AX)
    jnc gata_apel
    poz_cf:
    or word ptr [bp+4], 1 ; poz. imaginii din stivă a lui (CF)
gata_apel:
    pop bp ; refacere reg. salvate și a indicatorilor
    pop dx
    popf
    ret 2
fact endp

```

Imaginea stivei pentru această procedură este următoarea:

```

bp → bp_vechi
    dx_vechi
    indicatori
    adr_rev_procedură
    1
bp → ...
    N-1
bp → bp_vechi
    dx_vechi
    indicatori
    adr_rev_prog_principal
    N

```

2c) Procedura recursivă de calcul al factorialului, primește valoarea N prin stivă, și returnează rezultatul, N!, tot prin stivă.

```

facts proc
    cld ; inițializare (CF=0), la intrarea în procedură, sau
        ; în prog. principal, înainte de apelul procedurii
    pushf ; salvare indicatori
    push ax ; și registrele de lucru
    push dx ; afectat de înmulțire
    push bp
    mov bp, sp ; iniț. (BP), pentru a lua, la revenire, K-ul
    mov ax, [bp+10] ; citește valoarea lui N, din stivă
    cmp ax, 1 ; test sfârșit apel recursiv
    je gata_apel
    dec ax ; se trece la elementul următor, N-1
    push ax ; se depune acest termen în stivă
    call facts ; apelul recursiv
    pop ax ; extrage ultim. factorial, calculat, din stivă
    jc poz_cf
    mul word ptr [bp+10] ; rezultatul înmulțirii, K*f(K-1)
    jnc gata_apel ; este în (DX,AX)

```

```

poz_cf: or word ptr [bp+6], 1
gata_apel: mov [bp+10], ax      ; dep. rezultat în stivă
          pop  bp
          pop  dx
          pop  ax
          popf
          ret
facts  endp

```

Imagine stivă:

```

          (N - 1) <- (N-1)!
bp  →    bp_vechi
          dx_vechi
          ax_vechi
          indicatori
          adr_revenire_prog_principal
          N <- N!

```

3a) Proc. pentru afișarea unei valori din registrul AX în zecimal.

```

conv_bin_zec proc
;   procedura primește valoarea de convertit în registrul AX
  pushf ; se salvează indicatorii și registrele de lucru
  push cx
  push dx
  mov cx, 10 ; împărțitorul, utilizat pentru conversia
  xor dx, dx ; binar → zecimal, (DX, AX) = deîmpărțit
  div cx ; resturile reprezintă cifrele în zecimal
  mov cx, dx ; test condiție de sfârșit: câtul și restul sunt 0 ?
  or cx, ax
  je gata_apel ; dacă câtul și restul sunt 0, nu mai pun în stivă
  push dx ; depunere rest în stivă
  call conv_bin_zec ; dacă câtul <> 0, apelez recursiv proc.
  pop dx ; citire rest din stivă
  add dl, 30h ; conversie la codul ASCII
  mov ah, 2 ; apel funcție 2, de tipărire caracter
  int 21h
gata_apel:
  pop dx ; refacerea registrelor de lucru
  pop cx
  popf ; refacerea indicatorilor
  ret
conv_bin_zec endp

```

Configurația stivei pentru acest apel este următoarea:

```

          adr_rev_procedură
sp  →    restul (DX)

```

dx_vechi
cx_vechi
indicatori
adr_rev_prog_principal