Programiranje 1

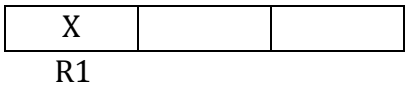
Aleksandar Veljković

URM - Rešenja zadataka

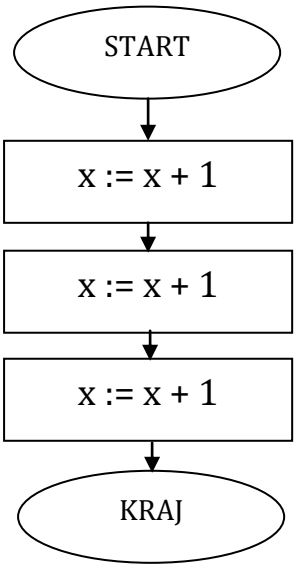
1. **Zadatak** Napisati URM program koji izračunava funkciju
$$f(x) = x + 3.$$

Rešenje
Izračunavanje funkcije svodi se na tri puta ponovljenu naredbu S nad ćelijom R1. Povratna vrednost ostaje u ćeliji R1.

Početna konfiguracija:



R1



Program

1. S(1) *x := x + 1*
2. S(1) *x := x + 1*
3. S(1) *x := x + 1*
4. J(1,1,100) *Skok na nepostojeću instrukciju i kraj programa*

2. **Zadatak** Napisati URM program koji izračunava funkciju

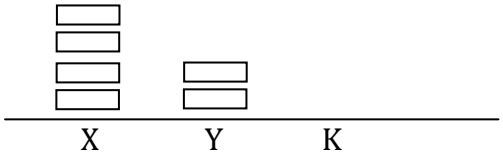
$$f(x,y) = \begin{cases} 1, & x \geq y \\ 0, & inače \end{cases}$$

Rešenje
Zadatak se rešava uvođenjem brojača **K** čija će se vrednost inicijalizovati na 0 i zatim u svakom koraku povećavati za jedan sve dok ne postane jednaka vrednosti **X** ili **Y**.

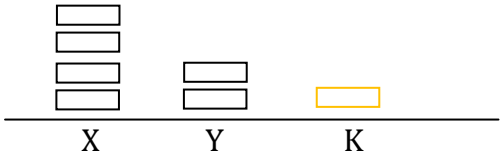
Ukoliko **K** dostigne prvo **Y** znači da je **Y manje ili jednako X** pa je time ispunjen prvi uslov funkcije i povratna vrednost će biti **1**. Ukoliko **K** nije dostiglo **Y**, ali jeste **X** znači da je **X manje od Y** i time je ispunjen drugi uslov funkcije pa je povratna vrednost programa **0**.

Primer x = 4, y = 2

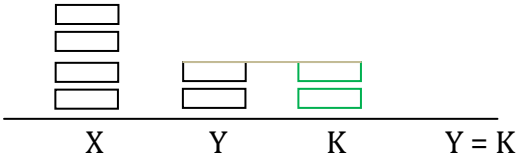
1.



2.

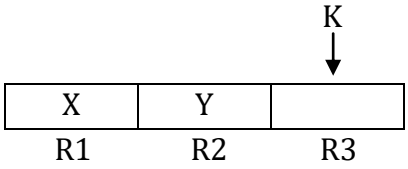


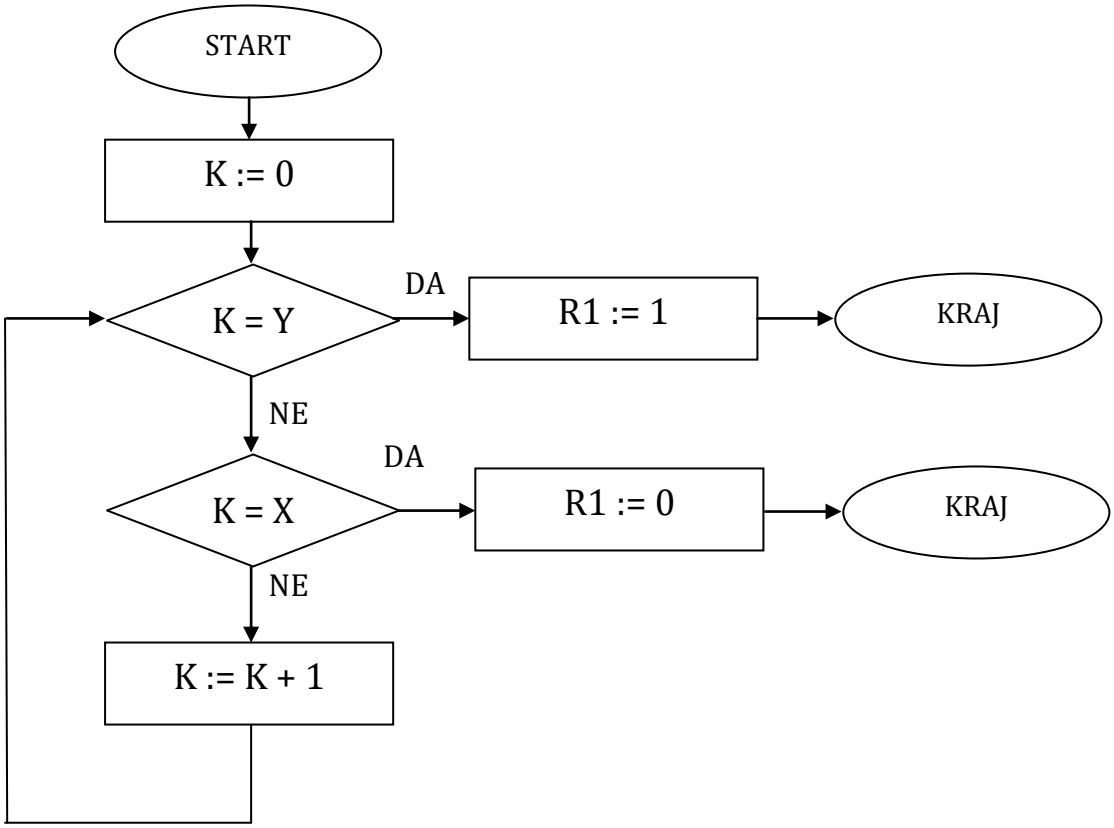
3.



Y = K => X ≥ Y => *f(x) = 1*

Početna konfiguracija:





Program

1. Z(3) *K := 0*
2. J(2,3,6) *K = Y ?*
3. J(1,3,9) *K = X ?*
4. S(3) *K := K + 1*
5. J(1,1,2) *Bezuslovni skok na naredbu 2. (petlja)*
6. Z(0) *R1 := 0*
7. S(1) *R1 := 1, f(x) = 1*
8. J(1,1,100) *Kraj*
9. Z(1) *R1 := 0, f(x) = 0*
10. J(1,1,100) *Kraj*

3. Zadatak Napisati URM program koji izračunava funkciju

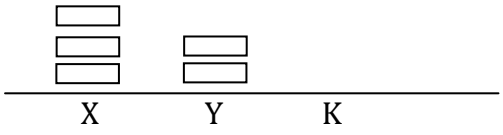
$$f(x, y) = x + y$$

Rešenje

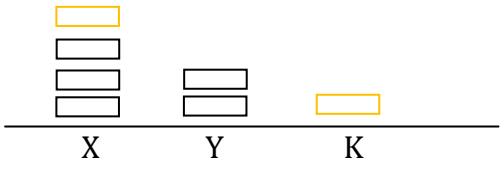
Ideja rešenja je dodavanje **Y jedinica** na **X jedinica**. Brojač **K** u svakoj iteraciji petlje označava broj jedinica koje su dodate na **X jedinica**. Tokom svake iteracije petlje proverava se da li je vrednost **K jednaka Y**, ukoliko jeste dodato je **Y jedinica** na **X** jedinica - ukoliko nije, vrednost brojača **K** se povećava za 1 i vrednost ćelije **R1** u kojoj se nalazi **X** se povećava za 1. Po završetku programa, vrednost ćelije **R1** biće zbir **X + Y**.

Primer x = 3, y = 2

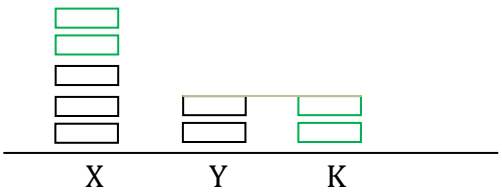
1.



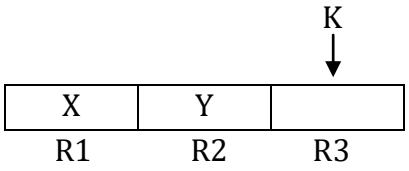
2.

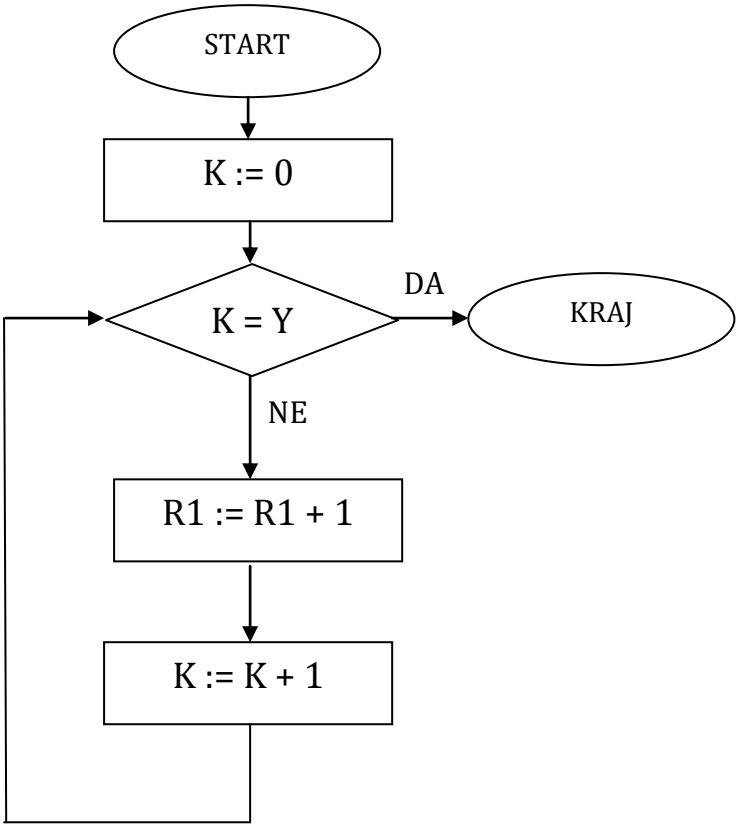


3.



Početna konfiguracija:





Program

1. Z(3) *K := 0*
2. J(2,3,100) *K = Y ? Ako jeste => Kraj*
3. S(1) *R1 := R1 + 1*
4. S(3) *K := K + 1*
5. J(1,1,2) *Bezuslovni skok na naredbu 2. (petlja)*

4. Zadatak Napisati URM program koji izračunava funkciju

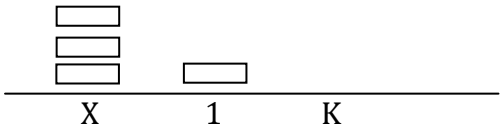
$$f(x) = x - 1$$

Rešenje

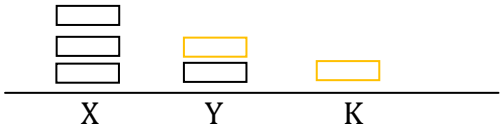
Slično kao u prethodnom zadatku, broji se koliko je jedinica potrebno dodati na **jedinicu** da bi ukupna vrednost bila jednaka **X**. U ćeliju **R2** upisujemo jedinicu a u **R3** će se nalaziti vrednosti brojača **K**. U svakoj iteraciji dodaje se po jedna jedinica na **R2** i na **K**. Kako je vrednost **R2** za jedan veća od vrednosti brojača **K**, u trenutku kada vrednost **R2** postane **jednaka vrednosti X**, vrednost brojača **K** biće **X - 1** i ostaje samo da se ta vrednost prebaci u ćeliju **R1**.

Primer x = 3

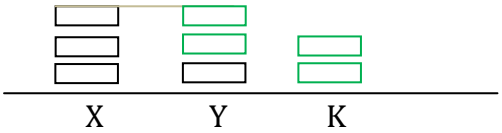
1.



2.



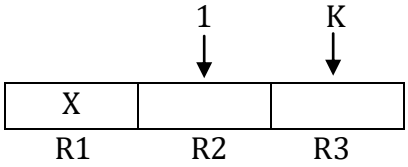
3.

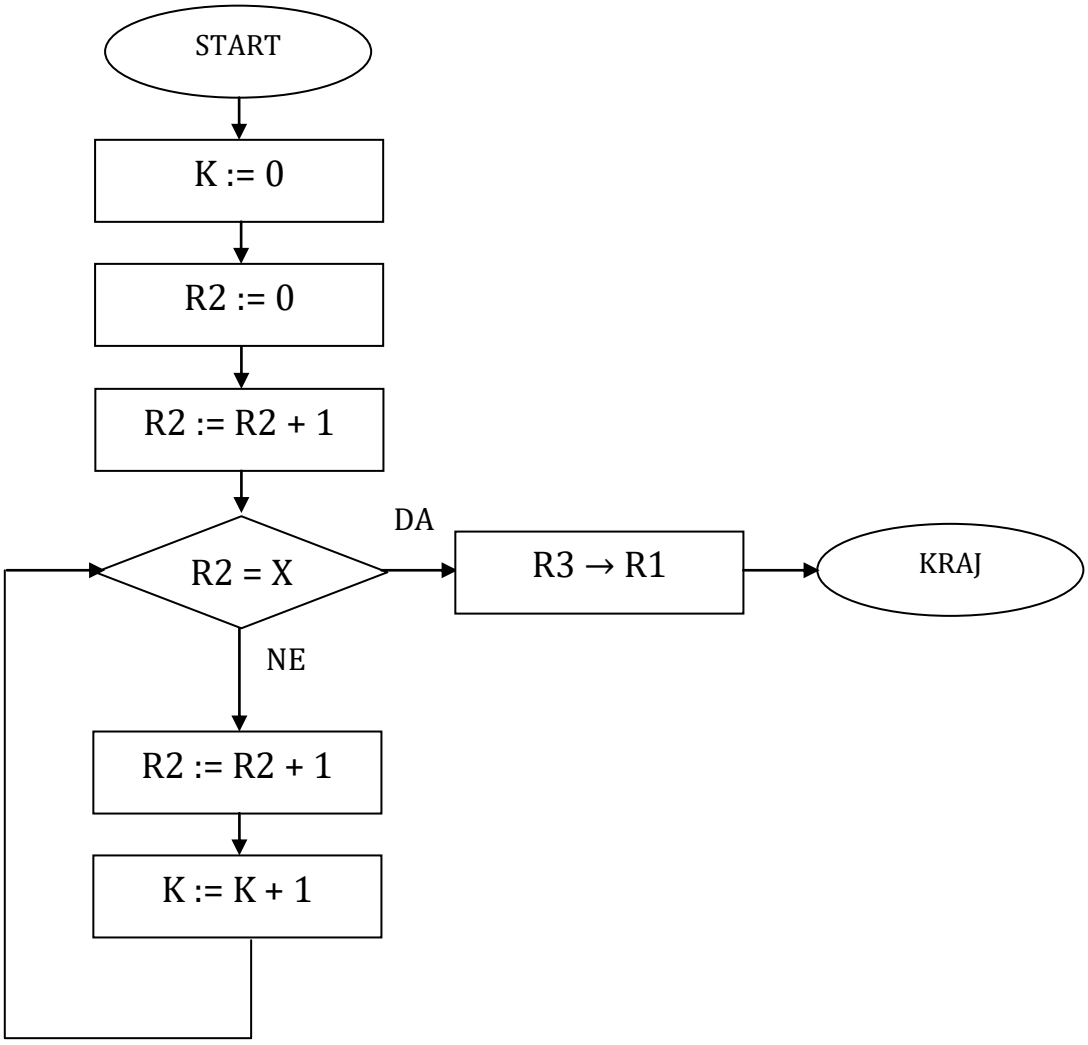


Program

1.	Z(3)	*K := 0*
2.	Z(2)	*R2 := 0*
3.	S(2)	*R2 := R2 + 1*
4.	J(1,3,8)	*R2 = X ?*
5.	S(2)	*R2 := R2 + 1*
6.	S(3)	*K := K + 1*
7.	J(1,1,4)	*Bezuslovni skok na naredbu 4.*
8.	T(3,1)	*K -> R1*
9.	J(1,1,100)	*Kraj*

Početna konfiguracija:





5. **Zadatak** Napisati URM program koji izračunava funkciju

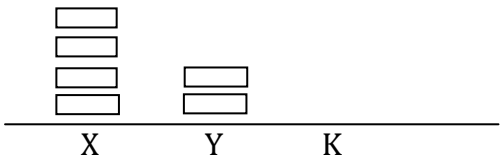
$$f(x, y) = x - y$$

Rešenje

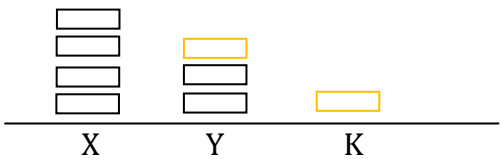
Slično kao u prethodnom zadatku, broji se koliko je jedinica potrebno dodati na **Y** da bi ukupna vrednost bila jednaka **X**. U ćeliji **R3** će se nalaziti vrednosti brojača **K**. U svakoj iteraciji dodaje se po jedna jedinica na **Y** i na **K**. Kako je vrednost **Y** za **Y** veća od vrednosti brojača **K**, u trenutku kada vrednost **Y+K** postane **jednaka vrednosti X**, vrednost brojača **K** biće upravo **X - Y** i ostaje samo da se ta vrednost prebaci u ćeliju **R1**.

Primer x = 4, y = 2

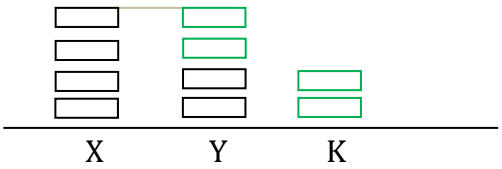
1.



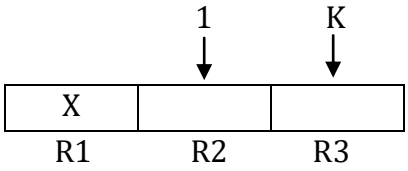
2.

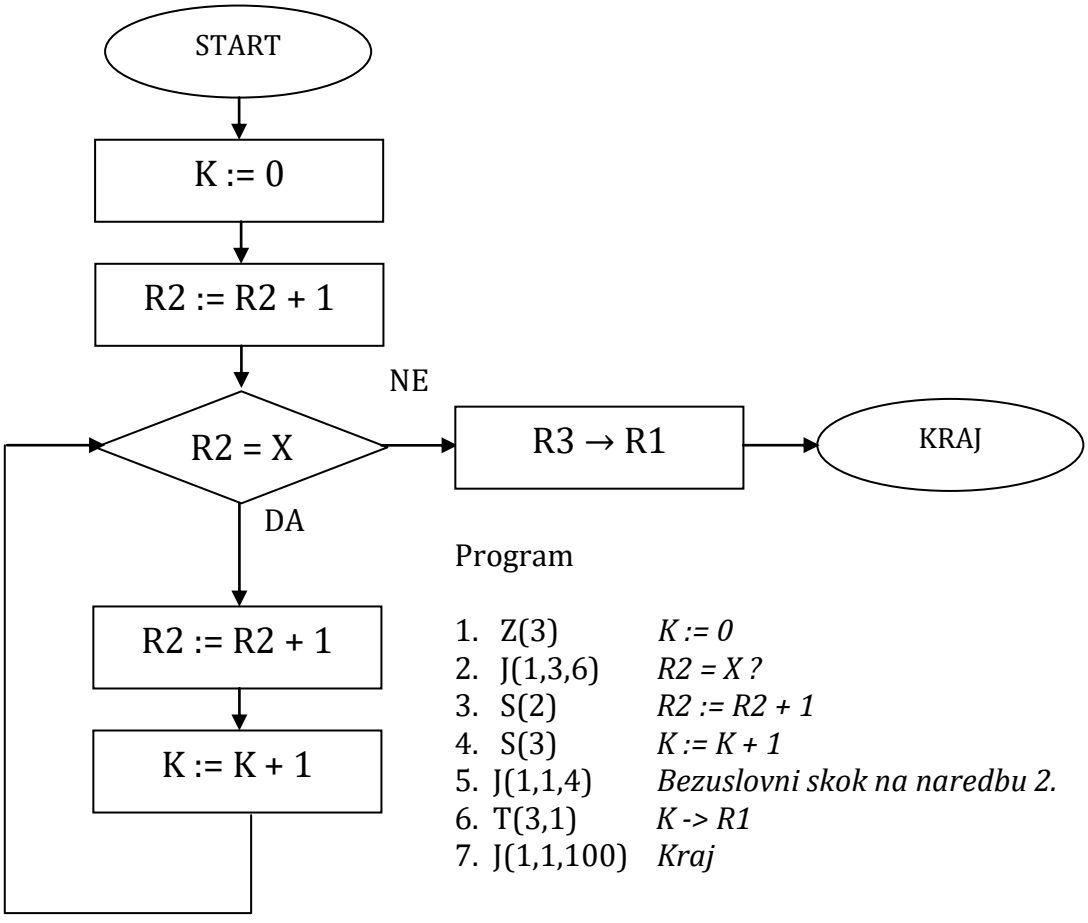


3.



Početna konfiguracija:





Program

1.	Z(3)	*K := 0*
2.	J(1,3,6)	*R2 = X ?*
3.	S(2)	*R2 := R2 + 1*
4.	S(3)	*K := K + 1*
5.	J(1,1,4)	*Bezuslovni skok na naredbu 2.*
6.	T(3,1)	*K -> R1*
7.	J(1,1,100)	*Kraj*

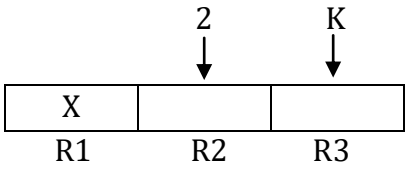
6. Zadatak Napisati URM program koji izračunava funkciju

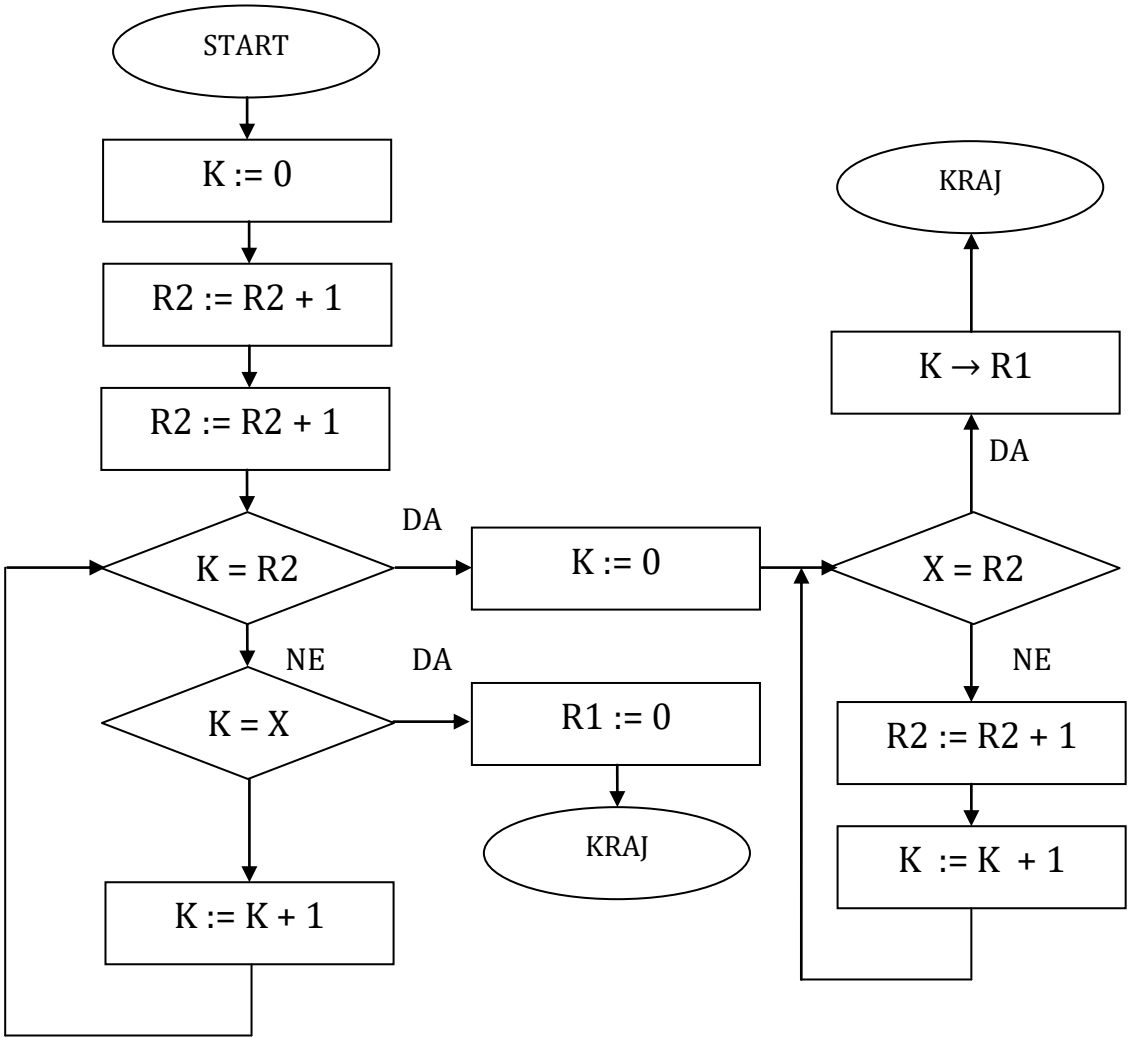
$$f(x) = \begin{cases} x - 2, & x \geq 2 \\ 0, & x < 2 \end{cases}$$

Rešenje

Prvi deo programa je provera uslova $\boldsymbol{x \geq 2}$ dok se u drugom delu (ukoliko je uslov ispunjen) izračunava razlika **X - 2**. **Napomena:** Zadatak je moguće uraditi kraće, tako da se istovremeno proverava uslov i izračunava razlika.

Početna konfiguracija:





Program

1. Z(3) *K := 0*
2. S(2) *R2 :=R2 + 1*
3. S(2) *R2 := R2 + 1 -Dvojka je napravljena*
4. J(2,3,) *K = R2 ? (K = 2)*
5. J(1,2,8) *K = X ?*
6. S(3) *K := K + 1*
7. J(1,1,4) *Bezuslovni skok na naredbu 4. (petlja)*
8. Z(1) *R1 := 0*
9. J(1,1,100) *Kraj*
10. Z(3) *K: = 0*
11. J(1,2,15) X = R2 ?
12. S(2) *R2 := R2 + 1*
13. S(3) *K := K + 1*
14. J(1,1,11) *Bezuslovni skok na naredbu 11. (petlja)*
15. T(3,1) *K -> R1*
16. J(1,1,100) *Kraj*

7. Zadatak Napisati URM program koji izračunava funkciju

$$f(x, y) = xy$$

Rešenje

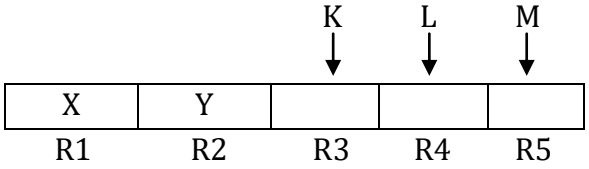
Množenje se svodi na sabiranje.

$$xy = x + x + \cdots + x = \underbrace{(1 + 1 + \cdots + 1)}_{\text{y puta}} + \underbrace{(1 + 1 + \cdots + 1)}_{\text{x puta}} + \cdots + \underbrace{(1 + 1 + \cdots + 1)}_{\text{x puta}}$$

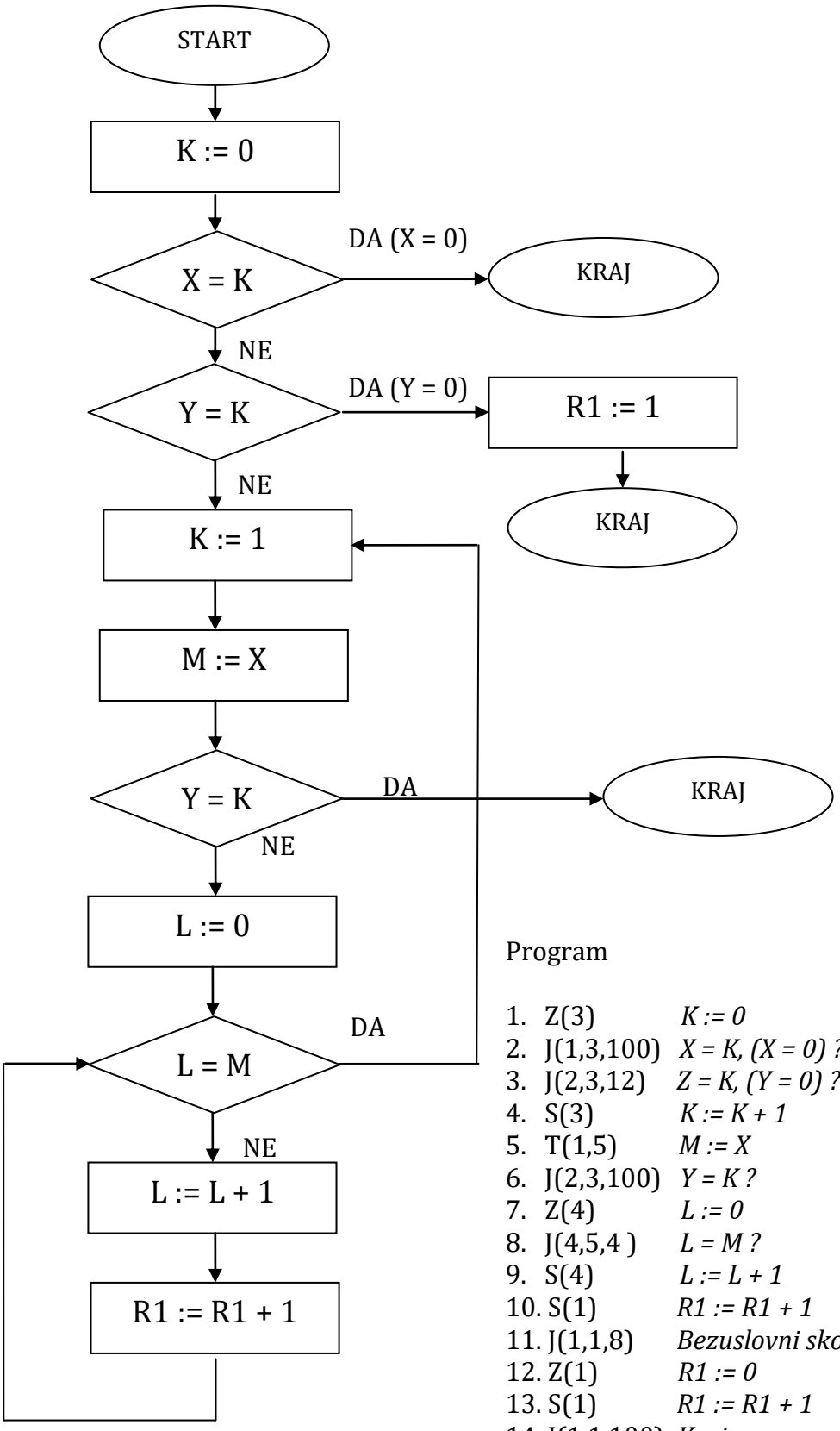
$$\underbrace{\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad}_{\text{x puta}}$$

y puta

Početna konfiguracija:



Brojač **K** broji koliko puta je dodata vrednost **X** na rezultat u **R1.** U svakoj iteraciji petlje vrednost **R1** je jednaka **K * X**. Kako je inicijalno vrednost **R1** jednaka **X** to je vrednost **K** inicijalno jednaka **1**. Brojač **L** učestvuje u formiranju svakog sabirka **X** i broji koliko je jedinica dodato za sabirak **X**. U **M** (R3) će se nalaziti kopija vrednosti **X** tako da prilikom svakog formiranja sabiraka **L** odbrojava do vrednosti iz **M**.



Program

1. Z(3) *K := 0*
2. J(1,3,100) *X = K, (X = 0) ? Ako jeste => Kraj*
3. J(2,3,12) *Z = K, (Y = 0) ?*
4. S(3) *K := K + 1*
5. T(1,5) *M := X*
6. J(2,3,100) *Y = K ?*
7. Z(4) *L := 0*
8. J(4,5,4) *L = M ?*
9. S(4) *L := L + 1*
10. S(1) *R1 := R1 + 1*
11. J(1,1,8) *Bezuslovni skok na naredbu 8.*
12. Z(1) *R1 := 0*
13. S(1) *R1 := R1 + 1*
14. J(1,1,100) *Kraj*

8. Zadatak Napisati URM program koji izračunava broj 1000 [1]

Rešenje

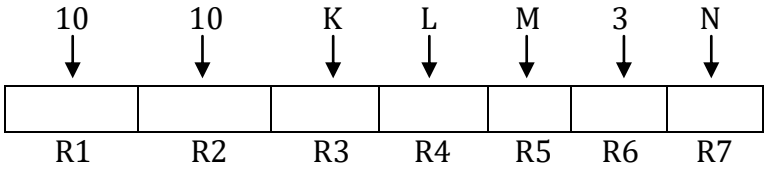
Stepenovanje se svodi na množenje a množenje na sabiranje.

$$1000 = 10^3 = 10 \cdot 10 \cdot 10$$

$$\mathbf{10 \cdot 10} \cdot 10 =$$
$$\mathbf{100} \cdot 10 =$$
$$\mathbf{1000}$$

Ideja algoritma je izvršiti množenje prva dva činioca, rezultat množenja upisivati na mesto prvog činioca i ponavljati postupak množenja sve do traženog stepena (3).

Početna konfiguracija:

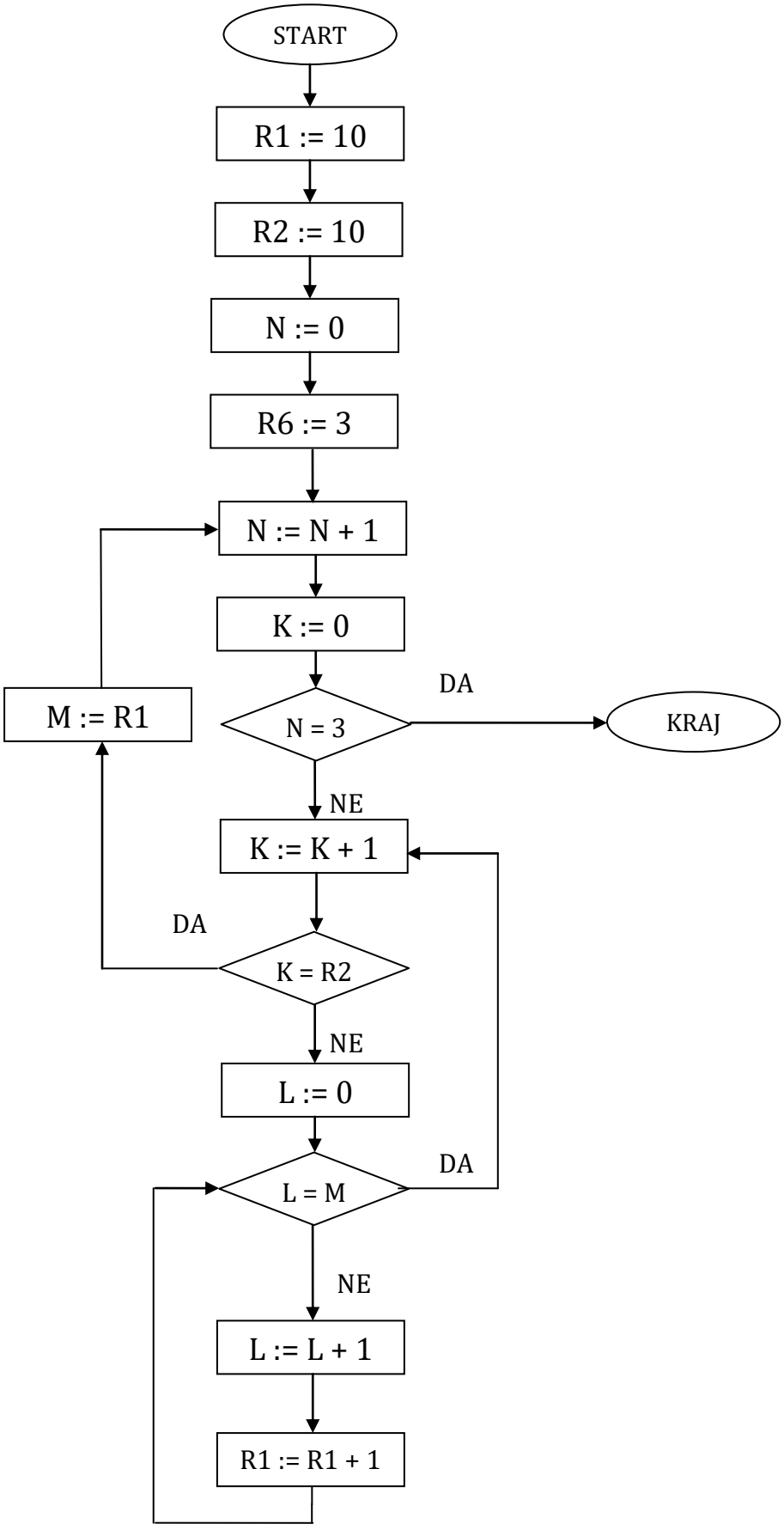


U **R1** se formira desetka a zatim vrednost **R1** kopira u **R2. R6** će čuvati stepen, vrednost **3**. Ćelije **R1** i **R2** će u toku algoritma predstavljati činioce. **K**, **L** i **M** će učestvovati u množenju baš kao što je slučaj u prethodnom zadatku. **N** će biti brojač koji odbrojava koliko puta je izvršeno množenje i zaustaviti proces kada dostigne vrednost **3** (brojač za stepen). Nakon svake iteracije množenja, rezultat množenja se upisuje u **R1**, na mesto prvog činioca i u **M**, za potrebe petlje koja dodaje jedinice.

Program

1. Z(1) *R1 := 0*
2. S(1) *R1 := R1 + 1*
3. S(1) *R1 := R1 + 1*
4. S(1) *R1 := R1 + 1*
5. S(1) *R1 := R1 + 1*
6. S(1) *R1 := R1 + 1*
7. S(1) *R1 := R1 + 1*
8. S(1) *R1 := R1 + 1*
9. S(1) *R1 := R1 + 1*
10. S(1) *R1 := R1 + 1*
11. S(1) *R1 := R1 + 1, (10 -> R1)*
12. T(1,2) *R2 := R1, 10 -> R2*
13. Z(7) *N := 0*
14. Z(6) *R6 := 3*
15. S(1) *R6 := R6 + 1*
16. S(1) *R6 := R6 + 1*
17. S(1) *R6 := R6 + 1, (3 -> R6)*
18. S(7) *N := N + 1*
19. Z(3) *K := 0*
20. J(7,6,100) *N = 3 ? Ako jeste => Kraj*
21. S(3) *K := K + 1*
22. J(2,3,28) *K = R2 ?*
23. Z(4) *L := 0*
24. J(4,5,21) *L = M ?*
25. S(4) *L := L + 1*
26. S(1) *R1 := R1 + 1*
27. J(1,1,24) *Bezuslovni skok na naredbu 24.*
28. T(1,5) *M := R1*
29. J(1,1,18) *Bezuslovni skok na naredbu 18.*

[1] Rešenje oblika 1000 puta napisana naredba S(1) se ne priznaje niti upisivanje broja 1000 u početnoj konfiguraciji.


START
R1 := 10
R2 := 10
N := 0
R6 := 3
N := N + 1
K := 0
N = 3
DA
KRAJ
NE
M := R1
K := K + 1
DA
K = R2
NE
L := 0
L = M
DA
NE
L := L + 1
R1 := R1 + 1

9. Zadatak Napisati URM program koji izračunava funkciju

$$f(x, y, z) = 2 \cdot (x + y + z)$$

Rešenje

Prvi deo rešenja predstavlja računanje zbira x + y + z i to kao sabiranje broja z sa zbirom x + y. Drugi deo podrazumeva kopiranje dobijenog zbira u pomoćnu ćeliju i sabiranje istog sa samim sobom. Drugi način je množenje zbira x + y + z brojem 2. Zadatak je samostalno urađen na času.

10. Zadatak Napisati URM program koji izračunava funkciju

$$f(x) = 2^x$$

Zadatak ostavljen za domaći