

Prevodioci i interpretatori - februar 2007. - praktični deo ispita

1. Datoteka sadrži opis relacija na konačnom skupu $\{1, 2, \dots, n\}$. Prvi deo (**domain_size**) sadrži veličinu domena n , drugi deo (**relations**) sadrži definicije relacija u obliku skupova uređenih parova, a treći deo (**queries**) sadrži niz logičkih formula bez promenljivih. Atomička formula $r(x, y)$ je tačna ako i samo ako je zadata relacija čije je ime r koja sadrži uređen par (x, y) . Formule se grade od atomičkih formula primenom operatora **and**, **or**, **not**. Napisati **perl** skript koji ispisuje istinitosnu vrednost svih navedenih logičkih formula. [18]

```
domain_size {
    3
}
relations {
    r := {(1, 3), (1, 2), (2, 1), (3, 3)};
    s := {(1, 1), (2, 2), (3, 3)};
}
queries {
    r(1, 1) and r(2, 2) and r(3, 3);                FALSE
    (r(1, 1) or r(2, 1)) and (not s(1, 2));          TRUE
}
```

2. Napisati interpretator za jezik koji ima elemente programskog jezika BASIC. Program se sastoji od niza linija obeleženih brojevima. Interpretator se pokreće kada se naiđe na **RUN** i interpretacija počinje od linije sa najmanjim brojem.

- (a) Naredba **PRINT** ispisuje nisku karaktera koja je navedena kao argument. Naredbom **GOTO** se vrši bezuslovni skok na liniju označenu datim brojem. [4]

```
10 PRINT "Hello"                Hello
20 GOTO 10                      Hello
RUN                             ....
```

Prevodioci i interpretatori - februar 2007. - praktični deo ispita

1. Datoteka sadrži opis relacija na konačnom skupu $\{1, 2, \dots, n\}$. Prvi deo (**domain_size**) sadrži veličinu domena n , drugi deo (**relations**) sadrži definicije relacija u obliku skupova uređenih parova, a treći deo (**queries**) sadrži niz logičkih formula bez promenljivih. Atomička formula $r(x, y)$ je tačna ako i samo ako je zadata relacija čije je ime r koja sadrži uređen par (x, y) . Formule se grade od atomičkih formula primenom operatora **and**, **or**, **not**. Napisati **perl** skript koji ispisuje istinitosnu vrednost svih navedenih logičkih formula. [18]

```
domain_size {
    3
}
relations {
    r := {(1, 3), (1, 2), (2, 1), (3, 3)};
    s := {(1, 1), (2, 2), (3, 3)};
}
queries {
    r(1, 1) and r(2, 2) and r(3, 3);                FALSE
    (r(1, 1) or r(2, 1)) and (not s(1, 2));          TRUE
}
```

2. Napisati interpretator za jezik koji ima elemente programskog jezika BASIC. Program se sastoji od niza linija obeleženih brojevima. Interpretator se pokreće kada se naiđe na **RUN** i interpretacija počinje od linije sa najmanjim brojem.

- (a) Naredba **PRINT** ispisuje nisku karaktera koja je navedena kao argument. Naredbom **GOTO** se vrši bezuslovni skok na liniju označenu datim brojem. [4]

```
10 PRINT "Hello"                Hello
20 GOTO 10                      Hello
RUN                             ....
```

- (b) Naredba INPUT omogućuje unos niske sa standardnog ulaza. Niska se smešta u promenljivu čije je ime navedeno. Imena promenljivih koje čuvaju niske karaktera se završavaju karakterom \$. Niske karaktera se mogu nadovezivati korišćenjem operatora +. [6]

```
10 INPUT "What is your name: "; name$           What is your name: Pera
20 PRINT "Hello, " + name$                       Hello, Pera
RUN
```

- (c) Naredba LET je naredba dodele. Funkcije LEFT\$ i RIGHT\$ omogućavaju izdvajanje prefiksa, odnosno sufiksa niske date dužine. [6]

```
10 LET text$ = "This is a test"
20 prefix$ = LEFT$(text$, 4);
30 suffix$ = RIGHT$(text$, 4);
40 PRINT prefix$                                This
50 PRINT suffix$                                test
RUN
```

- (d) Naredba IF-THEN je naredba uslovnog grananja. Operator = predstavlja operator poređenja niski. [6]

```
10 INPUT "What is your name: "; name$           What is your name:
15 IF name$ = "" THEN GOTO 10                   What is your name: Pera
20 PRINT "Hello, " + name$                       Hello, Pera
RUN
```

- (e) Petlje se ostvaruju korišćenjem FOR-NEXT. [10]

```
20 FOR i = 1 TO 2                                I: 1
30 FOR j = 1 TO 2                                J: 1
40 PRINT "I: " + i                               I: 1
50 PRINT "J: " + j                               J: 2
60 NEXT j                                         I: 2
70 NEXT i                                         J: 1
RUN                                              I: 2
                                              J: 2
```

- (b) Naredba INPUT omogućuje unos niske sa standardnog ulaza. Niska se smešta u promenljivu čije je ime navedeno. Imena promenljivih koje čuvaju niske karaktera se završavaju karakterom \$. Niske karaktera se mogu nadovezivati korišćenjem operatora +. [6]

```
10 INPUT "What is your name: "; name$           What is your name: Pera
20 PRINT "Hello, " + name$                       Hello, Pera
RUN
```

- (c) Naredba LET je naredba dodele. Funkcije LEFT\$ i RIGHT\$ omogućavaju izdvajanje prefiksa, odnosno sufiksa niske date dužine. [6]

```
10 LET text$ = "This is a test"
20 prefix$ = LEFT$(text$, 4);
30 suffix$ = RIGHT$(text$, 4);
40 PRINT prefix$                                This
50 PRINT suffix$                                test
RUN
```

- (d) Naredba IF-THEN je naredba uslovnog grananja. Operator = predstavlja operator poređenja niski. [6]

```
10 INPUT "What is your name: "; name$           What is your name:
15 IF name$ = "" THEN GOTO 10                   What is your name: Pera
20 PRINT "Hello, " + name$                       Hello, Pera
RUN
```

- (e) Petlje se ostvaruju korišćenjem FOR-NEXT. [10]

```
20 FOR i = 1 TO 2                                I: 1
30 FOR j = 1 TO 2                                J: 1
40 PRINT "I: " + i                               I: 1
50 PRINT "J: " + j                               J: 2
60 NEXT j                                         I: 2
70 NEXT i                                         J: 1
RUN                                              I: 2
                                              J: 2
```