U18CO018

Shekhaliya Shubham

PPL

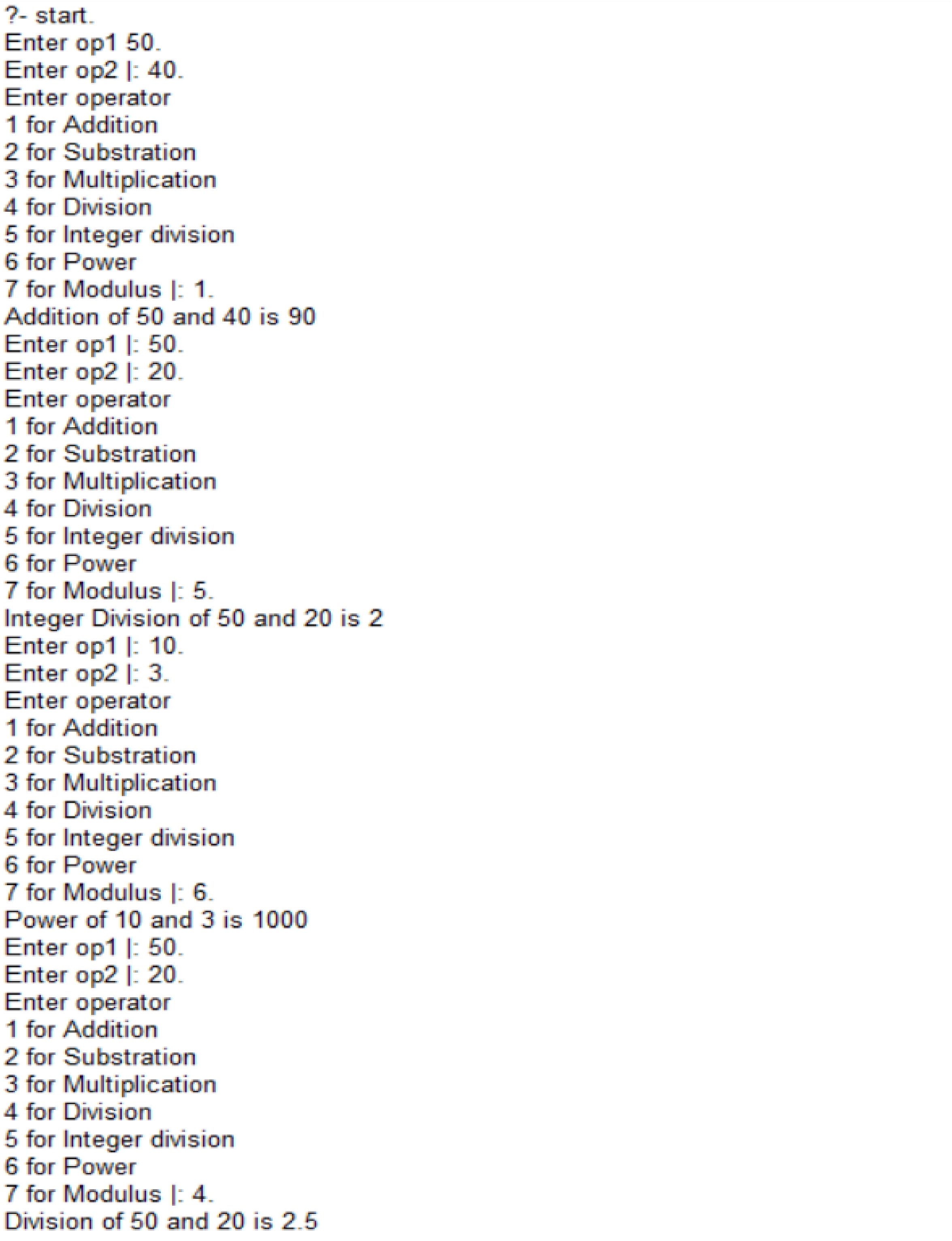
Lab Assignment 3

1. Write a prolog program to implement Menu Driven Calculator.

Code:

```
start:-
   write("Enter op1 "),
   read(X),
   write("Enter op2 "),
   read(Y),
   write("Enter operator"),nl,
   write("1 for Addition"),nl,
   write("2 for Substration"),nl,
   write("3 for Multiplication"),nl,
   write("4 for Division"),nl,
         write("5 for Integer division"),nl,
         write("6 for Power"),nl,
         write("7 for Modulus "),
   read(O),
   process(X,Y,O),
   start.

process(X,Y,O) :-
   O =:= 1,
   Z is X + Y,
   write('Addition of '), write(X), write(" and "), write(Y), write(" is "), write(Z), nl.

process(X,Y,O) :-
   O =:= 2,
   Z is X - Y,
   write('Substraction of '), write(X), write(" and "), write(Y), write(" is "), write(Z), nl.

process(X,Y,O) :-
   O =:= 3,
   Z is X * Y,
   write('Multiplication of '), write(X), write(" and "), write(Y), write(" is "), write(Z), nl.

process(X,Y,O) :-
   O =:= 4,
   Z is X / Y,
   write('Division of '), write(X), write(" and "), write(Y), write(" is "), write(Z), nl.

process(X,Y,O) :-
   O =:= 5,
   Z is X // Y,
   write('Integer Division of '), write(X), write(" and "), write(Y), write(" is "), write(Z), nl.

process(X,Y,O) :-
   O =:= 6,
   Z is X ** Y,
   write('Power of '), write(X), write(" and "), write(Y), write(" is "), write(Z), nl.

process(X,Y,O) :-
```

```
O =:= 7,

Z is X mod Y,

write('Modulus of '), write(X), write(" and "), write(Y), write(" is "), write(Z), nl.
```

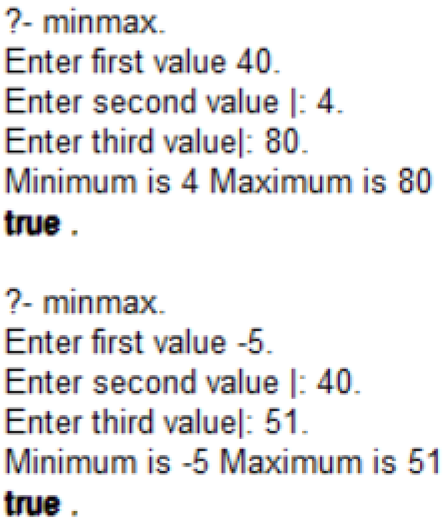


```
?- start.
Enter op1 50.
Enter op2 |: 40.
Enter operator
1 for Addition
2 for Substration
3 for Multiplication
4 for Division
5 for Integer division
6 for Power
7 for Modulus |: 1.
Addition of 50 and 40 is 90
Enter op1 |: 50.
Enter op2 |: 20.
Enter operator
1 for Addition
2 for Substration
3 for Multiplication
4 for Division
5 for Integer division
6 for Power
7 for Modulus |: 5.
Integer Division of 50 and 20 is 2
Enter op1 |: 10.
Enter op2 |: 3.
Enter operator
1 for Addition
2 for Substration
3 for Multiplication
4 for Division
5 for Integer division
6 for Power
7 for Modulus |: 6.
Power of 10 and 3 is 1000
Enter op1 |: 50.
Enter op2 |: 20.
Enter operator
1 for Addition
2 for Substration
3 for Multiplication
4 for Division
5 for Integer division
6 for Power
7 for Modulus |: 4.
Division of 50 and 20 is 2.5
```

2. Write a prolog program to find maximum and minimum of salaries of given 3 employees.

Code:

```
minmax:-
   write('Enter first value '),
   read(X),
   write('Enter second value '),
   read(Y),
   write('Enter third value'),
   read(Z),
   min(X, Y, Mina),
   min(Mina, Z, Min),
   max(X, Y, Maxa),
   max(Maxa, Z, Max),
   write('Minimum is '), write(Min), write(' Maximum is '), write(Max).

min(X, Y, X) :- X =< Y.
min(X, Y, Y) :- Y < X.

max(X, Y, X) :- X >= Y.
max(X, Y, Y) :- Y > X.
```

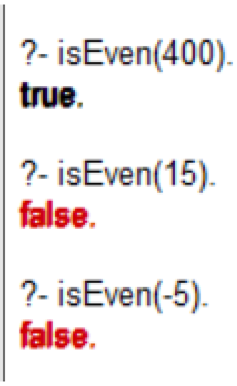


```
?- minmax.
Enter first value 40.
Enter second value |: 4.
Enter third value|: 80.
Minimum is 4 Maximum is 80
true .

?- minmax.
Enter first value -5.
Enter second value |: 40.
Enter third value|: 51.
Minimum is -5 Maximum is 51
true .
```

3. Write a prolog program to check whether given number is odd or even.

Code:

```
isEven(X):-
  Z is X mod 2,
  Z =:= 0.
```

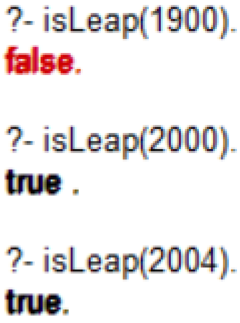


```
?- isEven(400).
true.

?- isEven(15).
false.

?- isEven(-5).
false.
```

4. Write a prolog program to check whether a given year is leap year or not.

Code:

```
isLeap(Y) :-

   Y mod 4 =:= 0,

   Y mod 100 =:= 0,

   Y mod 400 =:= 0.


isLeap(Y) :-

   Y mod 4 =:= 0,

   Z is Y mod 100,

   Z \= 0.
```



```
?- isLeap(1900).
false.

?- isLeap(2000).
true .

?- isLeap(2004).
true.
```

5. Write a prolog program to give grade to a student based on total marks given:

o 100 - 80 Grade A

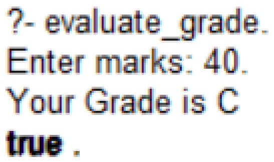
o 60 - 79 Grade B

o 35 - 59 Grade C

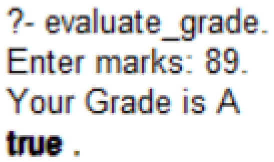
o 1 - 35 Grade D

Code:

```
evaluate_grade :-
  write("Enter marks: ") ,
  read(X),
  check(X,R),
  write("Your Grade is "),write(R), nl.

check( X , 'A') :- X =< 100, X >= 80.
check( X , 'B') :- X >= 60, X < 80.
check( X , 'C') :- X >= 35, X < 60.
check( X , 'D') :- X >= 1, X < 35.
```



```
?- evaluate_grade.
Enter marks: 40.
Your Grade is C
true .
```

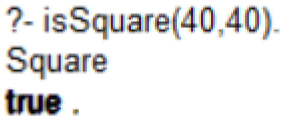


```
?- evaluate_grade.
Enter marks: 89.
Your Grade is A
true .
```

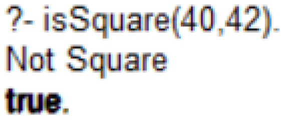
6. Write a prolog program to take values of length and breadth of a rectangle from user and check if it is square or not.

Code:

```
isSquare(X, Y):-
   X =:= Y,
   write('Square').

isSquare(X, Y):-
   X \= Y,
   write('Not Square').
```



```
?- isSquare(40,40).
Square
true .
```

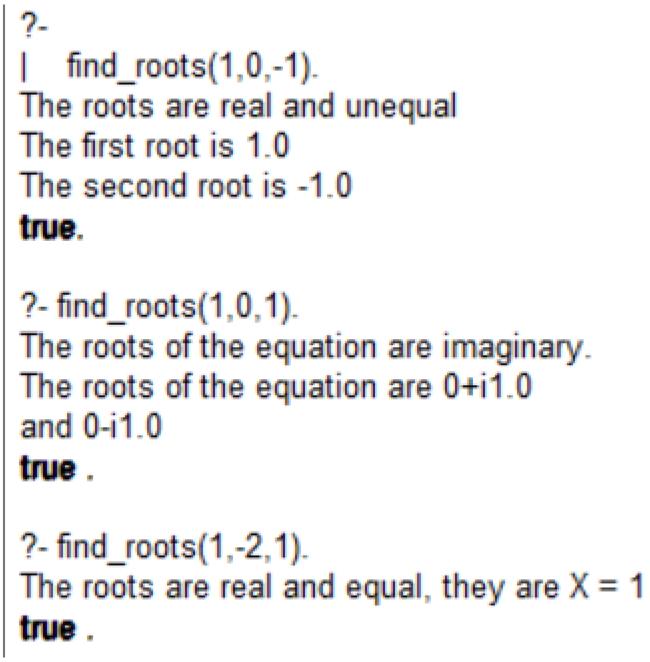


```
?- isSquare(40,42).
Not Square
true.
```

7. Write a PROLOG program to calculate the roots of quadratic equation Consider all possibilities real, equal, imaginary.

Code:

```
find_roots(A,B,C):-
   D = B*B - 4*A*C,
   type_of_root(A,B,D).

type_of_root(A,B,D):-
   D < 0,
   write("The roots of the equation are imaginary."), nl,
   T is -D,
   Z is sqrt(T),
   X is -B/(2*A),
   Y is Z/(2*A),
   write("The roots of the equation are "),write(X),write("+i"),write(Y),nl,
   write("and "), write(X), write("-i"), write(Y),nl.

type_of_root(A,B,D):-
   D =:= 0,
   X is -B/(2*A),
   write("The roots are real and equal, they are X = "), write(X), nl.

type_of_root(A,B,D):-
   D > 0,
   Z is sqrt(D),
   X1 is (-B + Z)/(2*A),
   X2 is (-B - Z)/(2*A),
   write("The roots are real and unequal"),nl,

   write("The first root is "), write(X1),nl,

   write("The second root is "), write(X2),nl.
```

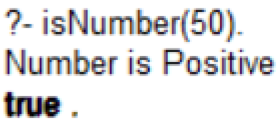


```
?-
|    find_roots(1,0,-1).
The roots are real and unequal
The first root is 1.0
The second root is -1.0
true.

?- find_roots(1,0,1).
The roots of the equation are imaginary.
The roots of the equation are 0+i1.0
and 0-i1.0
true .

?- find_roots(1,-2,1).
The roots are real and equal, they are X = 1
true .
```

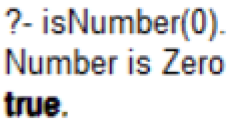
8. Write a PROLOG program to find the number whether the number is positive, negative or Zero.

Code:

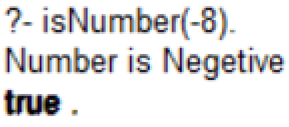
```
isNumber(X) :-
   X > 0,
   write('Number is Positive').

isNumber(X) :-
   X < 0,
   write('Number is Negetive').

isNumber(X) :-
   X =:= 0,
   write('Number is Zero').
```



```
?- isNumber(50).
Number is Positive
true .
```



```
?- isNumber(0).
Number is Zero
true.
```



```
?- isNumber(-8).
Number is Negetive
true .
```