

David Alejandro Velázquez Valdéz A01632648

Felix David De Haro Soto A01637589

Diego Ortiz Mariscal A01552000

Actividad 5.1 Programación Lógica

1. Write a predicate print every second/1 to print every other element in a list, beginning at the second element —i.e. the 2nd, 4th, 6th elements etc. It should always succeed provided it is given a list as its argument.

```
1 print_every_second([]).
2 print_every_second([_]).
3 Singleton variables: [X]
4 print_every_second([X,Y|_]):-
5 Singleton variables: [X]
6 write(Y),
7 print_every_second(Y).
```

```
print_every_second([1,2,3,4])
Singleton variables: [X]
Singleton variables: [X]
24
true

print_every_second([1,2,3,4,5,6])
Singleton variables: [X]
Singleton variables: [X]
246
true

print_every_second([1,2,3,4,5,6,7,8])
Singleton variables: [X]
Singleton variables: [X]
2468
true

print_every_second([])
Singleton variables: [X]
Singleton variables: [X]
true

print_every_second()
Singleton variables: [X]
Singleton variables: [X]
functor/3: Domain error: 'compound_non_zero_arity' expected, found 'print_every_second()'

?- print_every_second()
```

2. Write a predicate deconsonant/1 to print any element of a list that isn't a consonant (i.e. we want to print out the vowels fa,e,i,o,u). It should always succeed provided it is given a list as its argument (we assume that the input list only contains vowels and consonants).

```
1 deconsonant([]).
2 deconsonant([A|_]):-
3 vowel(A),
4 write(A),
5 deconsonant(_).
6 deconsonant([A|_]):-
7 deconsonant(_).
8 vowel(A),
9 vowel(e),
10 vowel(i),
11 vowel(o),
12 vowel(u).
```

```
deconsonant([a, e, c, d])
Singleton variables: [A]
or
true
Next 10 100 1,000 Stop

deconsonant([r, b, c, d])
Singleton variables: [A]
true

deconsonant([a, b, i, d])
Singleton variables: [A]
or
true
Next 10 100 1,000 Stop

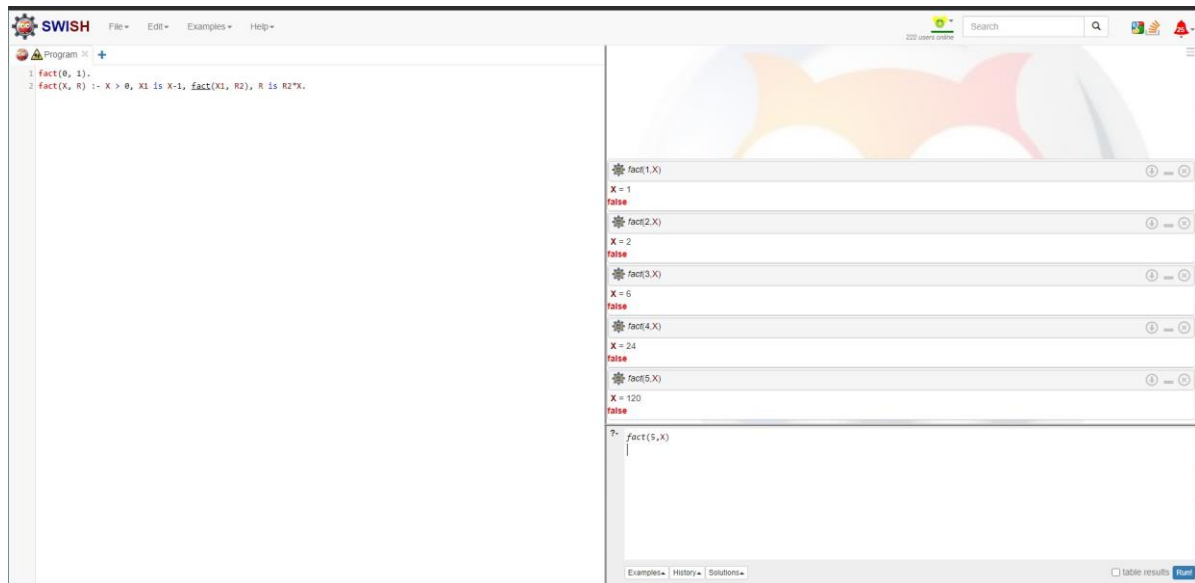
deconsonant([a, e, i, o])
Singleton variables: [A]
or
true
Next 10 100 1,000 Stop

?- deconsonant([a, e, i, o])
```

7. Write a predicate fact/2 which takes a natural number as first argument and returns the factorial of the number.

fact(0, 1).

fact(X, R) :- X > 0, X1 is X-1, fact(X1, R2), R is R2*X.



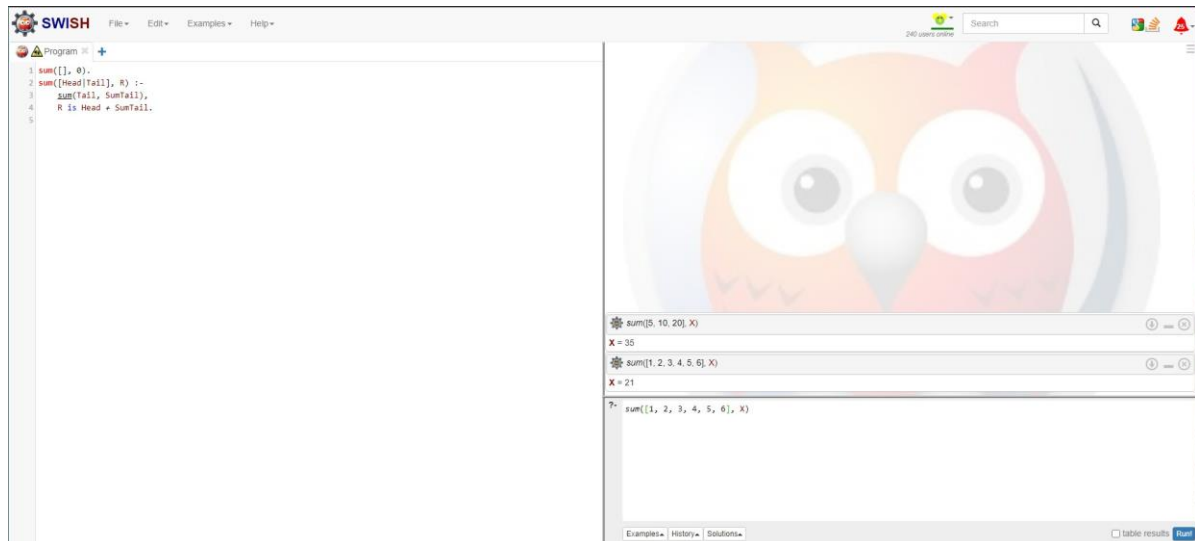
16. Define sum/2 to take a list of integers as input and return the output as their sum.

sum([], 0).

sum([Head|Tail], R) :-

sum(Tail, SumTail),

R is Head + SumTail.



19. Write a predicate `split/4` that splits a list into two parts, the length of the first part is given.

`split(L,0,[],L).`

`split([Head|TailX],N,[Head|TailY],List2) :- N > 0, N1 is N - 1, split(TailX,N1,TailY,List2).`

The screenshot shows the SWISH Prolog IDE interface. The editor on the left contains the following Prolog code:

```
1 split(L,0,[],L).
2 split([Head|TailX],N,[Head|TailY],List2) :- N > 0, N1 is N - 1, split(TailX,N1,TailY,List2).
```

The right-hand pane displays the results of several test queries:

- Query: `split([1,2,3,4],3,X,Y)`
X = [1,2,3],
Y = [4]
Result: **true**
- Query: `split([1,2,3,4,7,8,9,10],3,X,Y)`
X = [1,2,3],
Y = [4,7,8,9,10]
Result: **false**
- Query: `split([1,2,3,4,7,8,9,10],6,X,Y)`
X = [1,2,3,4,7,8],
Y = [9,10]
Result: **true**
- Query: `split([],0,X,Y)`
X = Y = []
Result: **true**

At the bottom of the right pane, the query `split([],0,X,Y)` is entered, and the 'Run' button is visible.