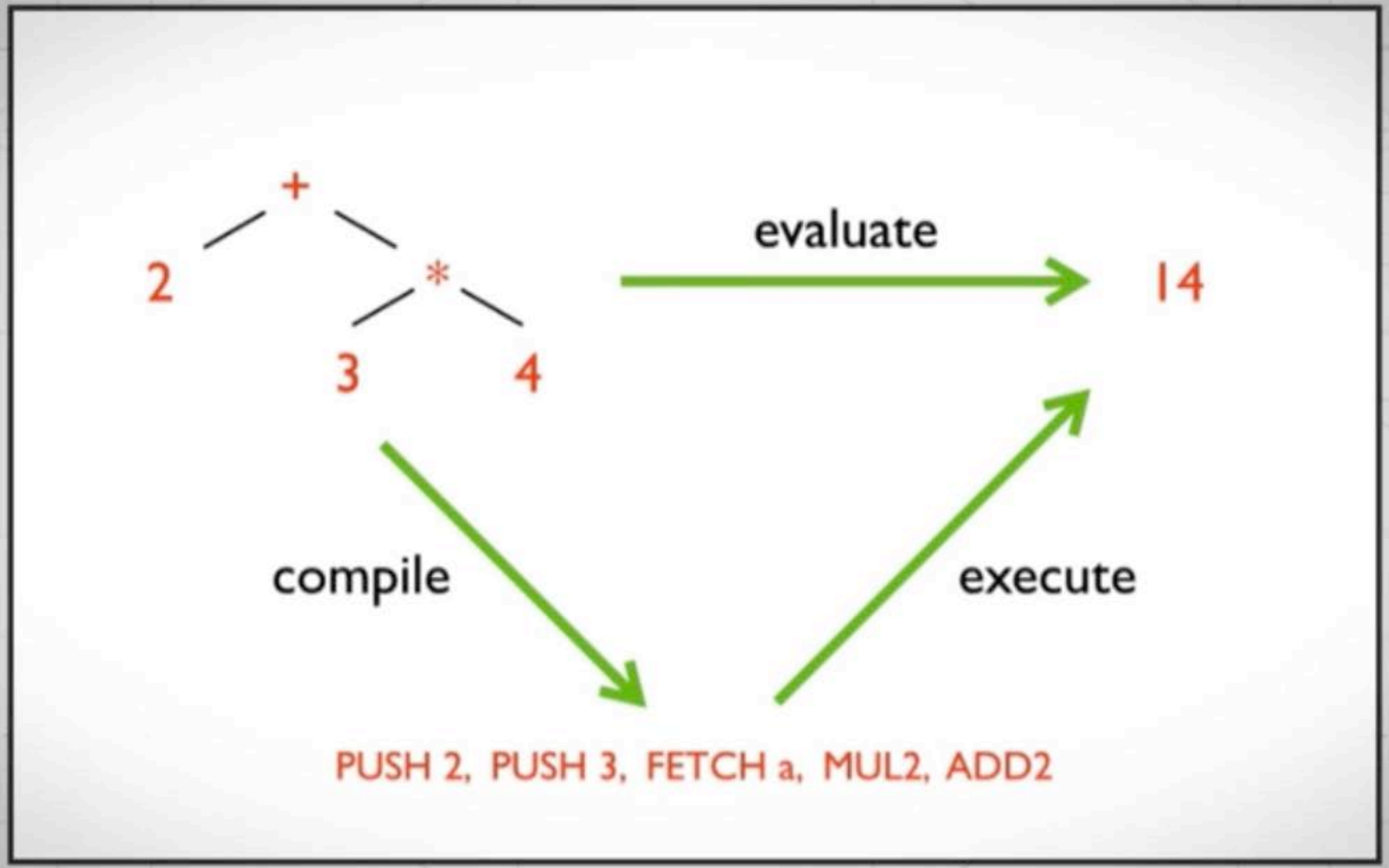
www.kent.ac.uk/elearning

ERLANG MASTER CLASS
LANGUAGE PROCESSING IN ERLANG

Video 04
EVALUATING EXPRESSIONS
with

SIMON THOMPSON
Professor of Logic & Computation, University of Kent


+
2
*
3
4
evaluate
14
compile
execute
PUSH 2, PUSH 3, FETCH a, MUL2, ADD2

University of Kent

Defining evaluation

Turn an expression into a number, by working out its value.

```
-spec eval(expr()) -> integer().
```

Defining evaluation

Turn an expression into a number, by working out its value.

```
-spec eval(expr()) -> integer().

eval({num,N}) ->
    ... ;
eval({add,E1,E2}) ->
    ... ;
eval({mul,E1,E2}) ->
    ... .
```

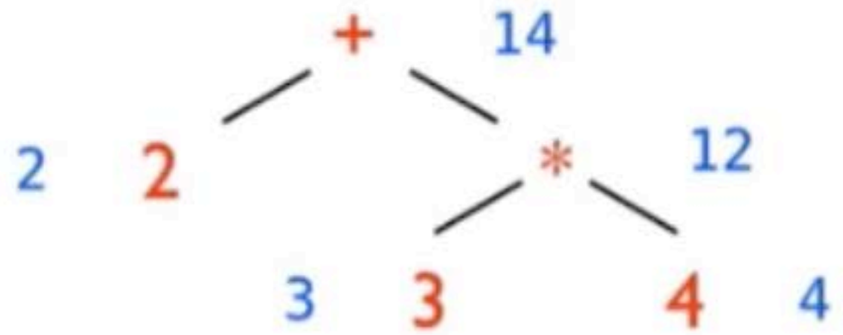
Defining evaluation

Turn an expression into a number, by working out its value.

```
-spec eval(expr()) -> integer().

eval({num,N}) ->
    N;
eval({add,E1,E2}) ->
    eval(E1) + eval(E2);
eval({mul,E1,E2}) ->
    eval(E1) * eval(E2).
```

Animating the calculation



Evaluation with variables

We need somewhere to `lookup` their values, e.g. `[{a,23},{v,17}]`

```
-type env() :: [{atom(),integer()}].

-spec eval(env(),expr()) -> integer().

eval(_Env,{num,N}) ->
    N;

eval(Env,{var,A}) ->
    lookup(A,Env);

eval(Env,{add,E1,E2}) ->
    eval(Env,E1) + eval(Env,E2);

eval(Env,{mul,E1,E2}) ->
    eval(Env,E1) * eval(Env,E2).
```

Evaluation with variables

We need somewhere to `lookup` their values, e.g. `[{a,23},{v,17}]`.

```
-type env() :: [{atom(),integer()}].

-spec lookup(atom(),env()) -> integer().

lookup(A,[{A,V}|_]) ->
    V;
lookup(A,[_|Rest]) ->
    lookup(A,Rest).
```

Fails when an atom not present e.g. `lookup(foo,[{a,23},{v,17}])`.

www.kent.ac.uk/elearning