ATIVIDADE 008: Introdução a Prog. Concorrente

Paradigmas de Linguagens de Programação

Fabrizio Honda
1715280006
Lic. em Computação

UEA
UNIVERSIDADE DO ESTADO DO AMAZONAS

DESCRIÇÃO DA ATIVIDADE

Explique o funcionamento do seguinte programa (Erlang):

```
-module(echo).
-export([go/0, loop/0]).

go() ->
Pid2 = spawn(echo, loop, [ ]),
Pid2 ! {self(), hello},
receive
{Pid2, Msg} ->
io:format("P1 ~w~n",[Msg])
end,
Pid2 ! stop.

loop() ->
receive
{From, Msg} ->
From ! {self(), Msg},
loop();
stop ->
end.
```

ERLANG

RASTREAMENTO DO CÓDIGO

progconc.erl

```
1  go()->
2      Pid2 = spawn(echo, loop, [])
3      Pid2 ! {self(), hello},
4      receive
5          {Pid2,Msg} ->
6          io:format("P1 ~w~n", [Msg])
7      end,
8      Pid2 ! stop
9
10 loop() ->
11     receive
12         {From, Msg} ->
13         From ! {self(), Msg},
14         loop();
15         stop ->
16             true
17     end.
```

go(): função principal (main) em Erlang.

spawn(echo, loop, [ ]): cria um novo processo, executa "loop" e envia uma lista vazia (pois esta função não recebe nada).

loop(): função definida no código.

Pid2 ! {self(), hello}: retorna a mensagem hello para a função loop após o spawn.

ERLANG

RASTREAMENTO DO CÓDIGO

progconc.erl

```
1  go()->
2      Pid2 = spawn(echo, loop, [])
3      Pid2 ! {self(), hello},
4      receive
5          {Pid2,Msg} ->
6          io:format("P1 ~w~n", [Msg])
7      end,
8      Pid2 ! stop
9
10 loop() ->
11     receive
12         {From, Msg} ->
13         From ! {self(), Msg},
14         loop();
15         stop ->
16             true
17     end.
```

receive: se o padrão recebido for igual a {From, Msg}, executa as próximas linhas de código

From ! {self(), Msg}: retorna a mensagem recebida para a função principal (go()).

loop(): em seguida, chama-se recursivamente até a condição de parada for satisfeita.

ERLANG

RASTREAMENTO DO CÓDIGO

progconc.erl

```
1  go()->
2      Pid2 = spawn(echo, loop, [])
3      Pid2 ! {self(), hello},
4      receive
5          {Pid2,Msg} ->
6          io:format("P1 ~w~n", [Msg])
7      end,
8      Pid2 ! stop
9
10 loop() ->
11     receive
12         {From, Msg} ->
13         From ! {self(), Msg},
14         loop();
15         stop ->
16             true
17     end.
```

receive: se o padrão recebido for igual a {Pid2, Msg}, executa as próximas linhas de código

io:format("P1 ~w~n", [Msg]): formata a saída do programa e imprime a mensagem recebida.

Pid2 ! stop: indica que o processo 2 pode encerrar-se, tornando verdadeira a condição de stop -> true.

ERLANG