

Introdução à Inteligência Artificial

Licenciatura em Engenharia Informática, Engenharia Informática – Pós Laboral e
Engenharia Informática – Curso Europeu

2º Ano – 1º semestre

2016/2017

Aulas Laboratoriais

Ficha 1: Introdução ao NetLogo

1. O que é o NetLogo?

O NetLogo (<http://ccl.northwestern.edu/netlogo>) é um ambiente de programação adequado à modelação de sistemas multi-agente. Permite realizar simulações de diferentes tipos, como por exemplo de ecossistemas ou de fenómenos sociais.

O NetLogo é baseado no conceito de agente: uma entidade autónoma, com características e regras próprias que especificam o seu comportamento. Tem a capacidade para interagir com o ambiente em que está inserido e de tomar as decisões que lhe permitam atingir os seus objectivos. O programador tem total liberdade para definir quais as características do agente, ao nível das percepções, acções ou estrutura interna (por exemplo, se tem ou não memória).

As características principais do NetLogo são:

- Os agentes actuam individualmente ao longo do tempo, tendo a capacidade de alterar o seu comportamento;
- Podem ser estabelecidas interacções directas ou indirectas entre os agentes;
- O ambiente é completamente definido pelo programador/utilizador;
- Possui ferramentas poderosas para estudo e análise dos resultados de simulação (durante e/ou no final das experiências).

Existem dois níveis de utilização do NetLogo:

- Interacção com modelos já existentes: o ambiente permite que um utilizador efectue simulações com modelos já existentes. Normalmente é possível especificar/alterar alguns parâmetros do modelo e verificar quais os efeitos produzidos ao nível de resultados;
- Concepção e desenvolvimento de modelos de simulação.

O NetLogo possui ainda três tipos de agentes (operando em simultâneo):

- **Patches** – células que permitem criar um ambiente 2D;
- **Turtles** – agentes que se movem no ambiente 2D criado pelas *patches*;
- **Observador** – agente que observa o ambiente de simulação constituído pelas *turtles* e pelas *patches*, e que pode atuar sobre ele.

2. Utilização do NetLogo

Além de poderem ser criados modelos de raiz, a biblioteca do NetLogo disponibiliza uma grande quantidade de modelos de simulação, prontos a utilizar. Esses modelos podem

também ser alterados, de acordo com as necessidades do utilizador. Assim, para se entender o modo de funcionamento do NetLogo, começa-se por aceder a um modelo existente (figura 1), fazendo o seguinte:

- Iniciar o NetLogo;
- No seu menu principal, escolher a opção **File** → **Models Library** (ou pressionar as teclas *Ctrl+M*);
- Na subjanela que aparecerá, seleccionar a pasta **Biology**;
- Seleccionar o modelo **Wolf Sheep Predation** e pressionar no botão **Open** (toda a explicação deste modelo pode ser encontrada no Tutorial #1, do manual do NetLogo, em <http://ccl.northwestern.edu/netlogo/docs/>);

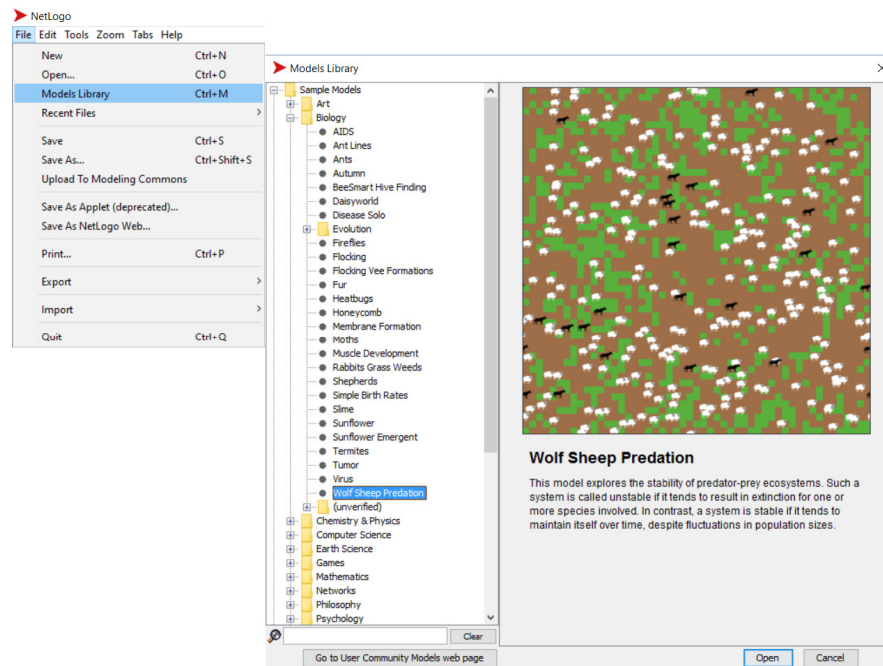


Figura 1 – Acesso a modelo do NetLogo (exemplo – o *Wolf Sheep Predation*).

Ao se abrir o modelo, pode-se verificar que ele é constituído pelas seguintes componentes:

- Área de simulação – ambiente onde se vão movimentar os agentes definidos no modelo (figura 2a);
- Diversos tipos de botões – para a especificação de parâmetros do modelo a simular no NetLogo e para a execução de certos procedimentos (figura 2b);
- Gráficos e monitores – para apresentação de resultados (figura 2c).

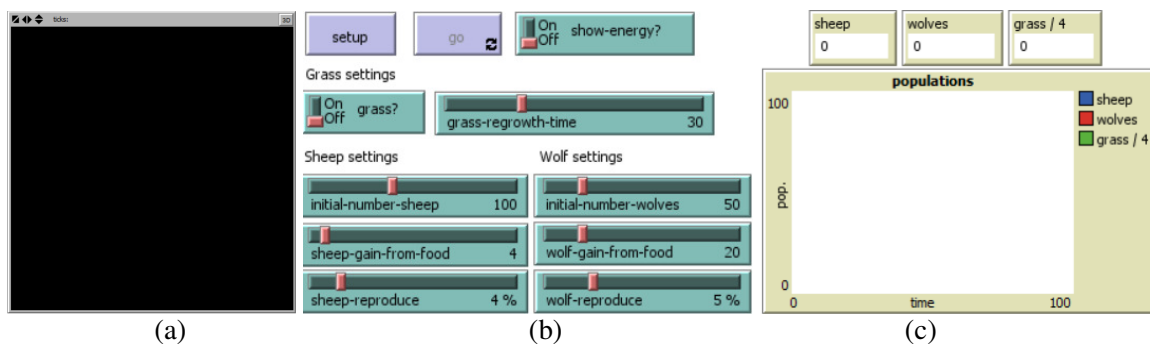


Figura 2 – (a) Área de simulação do NetLogo; (b) Exemplos de variados tipos de botões existentes num modelo do NetLogo; (c) Exemplos de gráficos e monitores existentes num modelo do NetLogo.

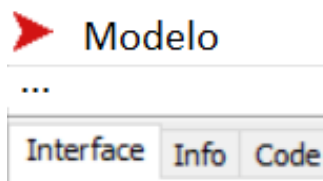


Figura 3 – Separadores de acesso aos ambientes de desenvolvimento do NetLogo.

Por fim é importante mencionar que para se criar um modelo no NetLogo tem que se passar pelos seguintes ambientes de desenvolvimento (figura 3):

- *Interface* – ambiente gráfico, onde se encontra a área de simulação, se adicionam botões, gráficos e monitores, entre outras coisas, e onde se pode visualizar o progresso da simulação, conforme se pode ver na figura 4;

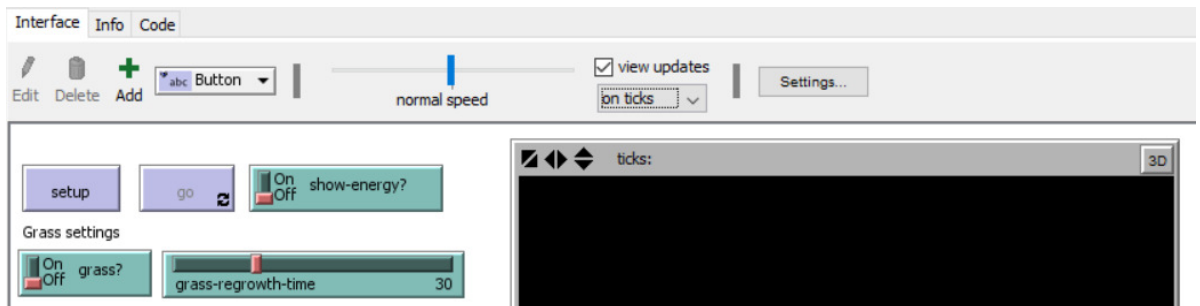


Figura 4 – Ambiente *Interface*.

- *Info* – ambiente de documentação, onde se pode ver e escrever informações sobre o modelo. Para se escrever essas informações deve-se pressionar o botão *Edit*, conforme se pode ver na figura 5;

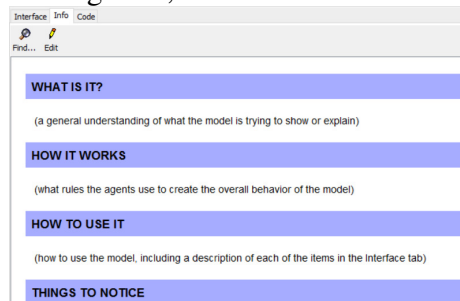


Figura 5 – Ambiente *Info*.

- *Code* – Ambiente de codificação, onde se pode ver e implementar código, escrito em linguagem Logo, que irá controlar a simulação ou o comportamento do modelo, conforme se pode ver na figura 6.

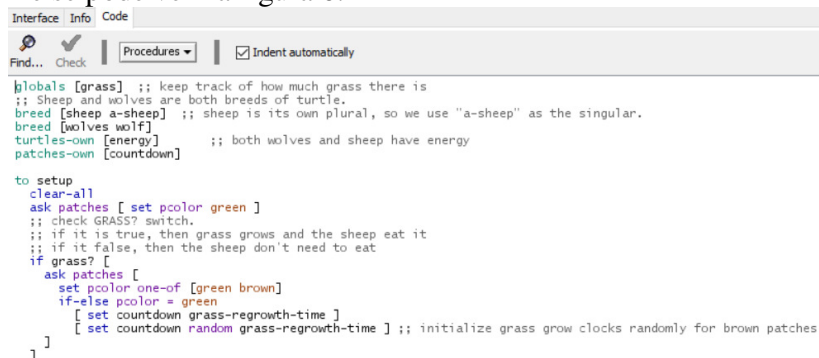


Figura 6 – Ambiente *Code*.

A área de simulação da figura 2(a) é o ambiente em que se movimentam os agentes. Este ambiente é constituído por uma grelha bidimensional de células. É possível configurar as dimensões deste ambiente. Para isso deve-se fazer o seguinte:

- Pressionar o botão direito do rato sobre a grelha;
- Selecionar a opção **Edit**, de forma a aparecer a figura 7;
- Alterar as coordenadas máximas para o eixo dos *XX* (*max-pxcor*) e dos *YY* (*max-pycor*).

A origem do ambiente de simulação é a célula central, que tem coordenadas (0, 0).

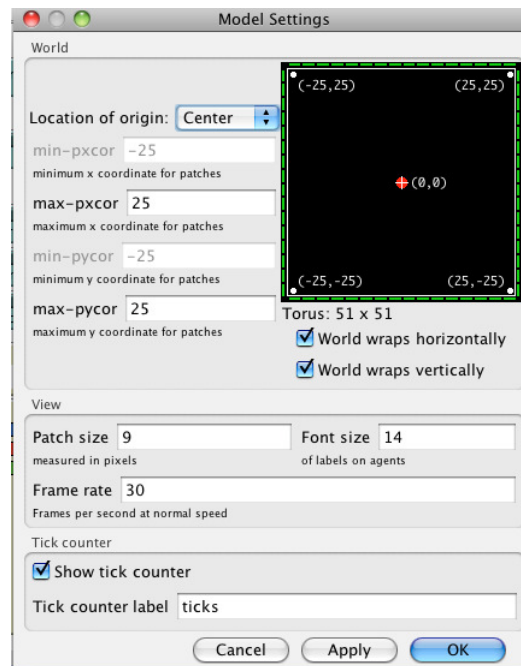


Figura 7 – Alteração da área de simulação do NetLogo.


Depois de se abrir ou criar um modelo tem que se **efetuar as simulações**. Normalmente as simulações dos modelos passam pelas seguintes fases, seguindo a respetiva ordem:


- Inicialização dos parâmetros;
- Execução cíclica de um conjunto de instruções que vai alterando esses parâmetros e efetuando a simulação pretendida.

Estas duas fases costumam estar retratadas nos botões a usar no ambiente *Interface*, os quais têm sempre uma ação associada (conjunto de instruções) e permitem controlar o modelo. Assim, para se definir, quando se pressiona um botão, se a ação a executar é cíclica ou não, utilizam-se os botões do tipo:

- *Once* – quando se pretende executar a ação apenas uma única vez;
- *Forever* – quando se pretende executar a ação repetidamente. Pode-se sempre parar a execução cíclica, pressionando novamente o botão que lhe deu início. Os botões *forever* têm duas setas no canto inferior direito, de forma a se distinguirem dos botões *Once*.

Para se verificar o que foi dito acima, realizar a simulação do modelo *Wolf Sheep Predation*, efetuando os seguintes passos:

- Pressionar o botão **setup** (), de forma a inicializar o modelo (definir quantas ovelhas e quantos lobos existem inicialmente no ambiente, qual a sua taxa de

- reprodução, limpar a área gráfica de simulação e colocar-lhe as *patches* com e sem relva, as ovelhas e os lobos, etc.);
- Pressionar o botão **go** (), de forma a ver como o ambiente se vai transformando ao longo do tempo;
 - Voltar a pressionar o botão **go** quando pretender parar a simulação;
 - Voltar a pressionar o botão **go** (continua do ponto onde foi parada) e deixar que a simulação termine por si.

Em qualquer simulação a fazer, a **recolha e apresentação de resultados** da simulação do modelo é muito importante. O NetLogo permite que os resultados vão sendo atualizados ao longo da simulação. Para isso, usa normalmente duas maneiras simples:

- Os gráficos;
- Os monitores.

Conforme se pode ver na figura 8, os gráficos aparece a evolução dos valores ao longo do tempo, enquanto nos monitores aparecem os valores atuais.

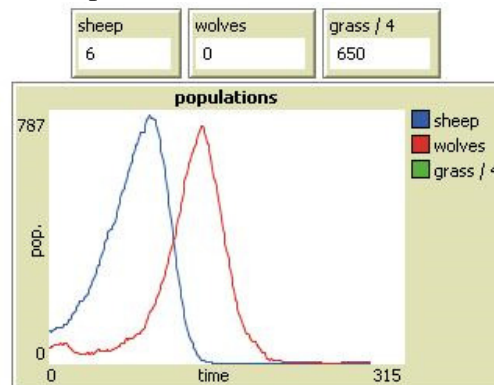


Figura 8 – Formas de recolha e apresentação de resultados da simulação do modelo.

A **parametrização do modelo** é feita mexendo em certos botões existentes que a área *Interface*. O NetLogo permite alterar os parâmetros associados a esses botões e que controlam o funcionamento dos modelos (antes e durante a simulação). No modelo do exemplo existem:

- **Switches** (interruptores) – que ativam/desativam um determinado parâmetro do modelo definido;
- **Sliders** (cursosores) – que atribuem um valor (de entre uma gama de valores possíveis) a um determinado parâmetro do modelo criado.

Para se confirmar o que foi dito acima:

- Identificar os parâmetros que podem ser configuráveis no modelo (consultar a área *Info* para esclarecer eventuais dúvidas);
- Efetuar alterações nos parâmetros e verificar quais os efeitos nos resultados da simulação;
- Procurar encontrar conjuntos de parâmetros que garantam o equilíbrio do ecossistema.

As características e os comportamentos dos componentes do modelo são programados através de um conjunto de procedimentos (funções). É, no entanto, possível recorrer ao **centro de comando**, apresentado na figura 9, para efetuar pequenos ajustes, a qualquer

altura da simulação, através de comandos dirigidos a qualquer tipo de agente. Para se perceber melhor a utilização do centro de comando:

- Iniciar uma simulação e executar um de cada vez os seguintes comandos (os comandos devem ser escritos na linha que está por baixo da janela principal do centro de comando – ver figura 9):
 - ask patches [set pcolor yellow]
 - ask turtles [set color 95]
 - setup
 - ask sheep [set color pink]
 - ask turtles [die]
- Tente interpretar os comandos que foram efectuados, nomeadamente no que diz respeito aos agentes que são afectados por eles.

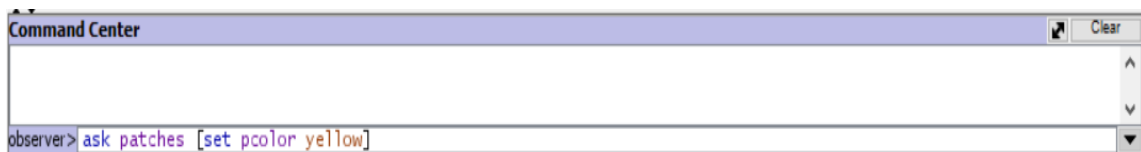


Figura 9 – Centro de comando do NetLogo.

É possível consultar as características de um agente através do seu monitor individual de forma a poder ser feita uma **análise detalhada desse agente**. O monitor de um agente (que pode ser uma *patch* ou uma *turtle*) pode ser obtido pressionando o botão direito do rato diretamente sobre ele, conforme se pode ver na figura 10a. É na janela que, entretanto, aparecerá depois de ser feita a ação anterior (figura 10b), que as características do agente podem ser consultadas ou alteradas.

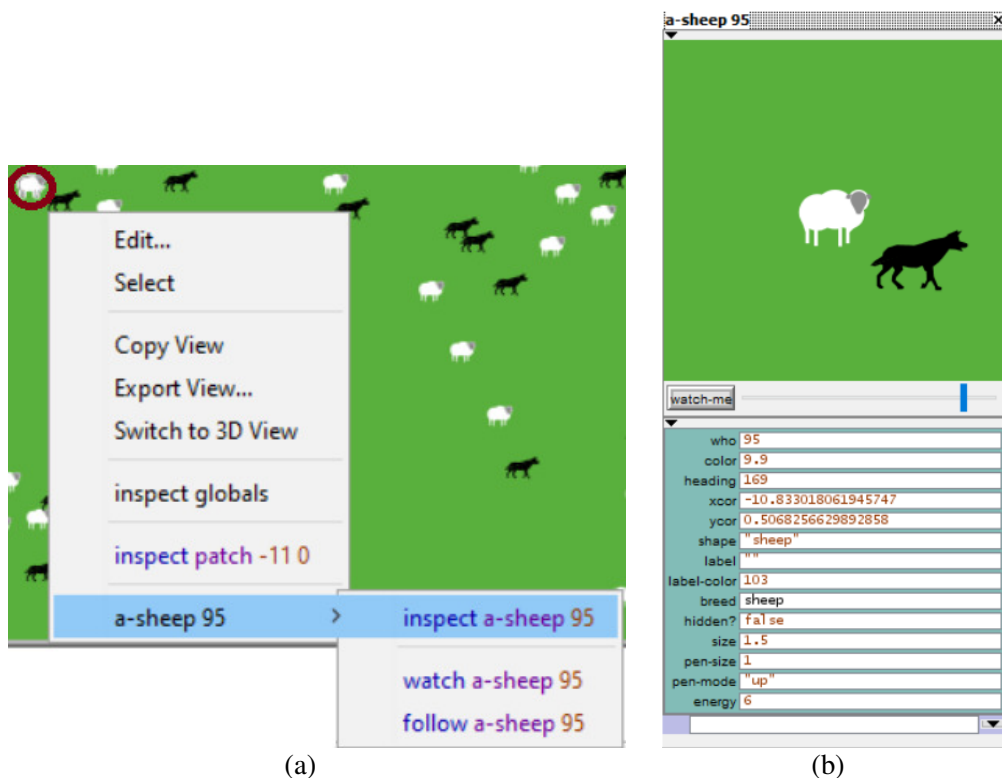


Figura 10 – (a) Forma de consulta das características do agente *a-sheep 95*; (b) Janela com as características do agente *a-sheep 95*.

Por fim, a aprendizagem sobre o modo de funcionamento do NetLogo tem que passar pela **análise do código implementado**. Assim, seguir as seguintes tarefas:

- Ir para o separador *Code*;
- Analisar o código implementado, tentando decifrar o que faz cada procedimento/instrução e anotar as instruções que achar mais interessantes para utilização em futuros exercícios. Como exemplo, tentar encontrar no código as instruções que fazem as seguintes tarefas?
 - a) Criar dois tipos de *turtles*, neste caso lobos e ovelhas;
 - b) Criar o número de lobos e ovelhas estipulados nas variáveis correspondentes;
 - c) Reprodução dos lobos: como se cria o 2º lobo, qual a energia com que fica?
 - d) Morte dos lobos ou ovelhas;
 - e) Comer a erva, o que acontece à energia de quem come, o que acontece à *patch* onde estava a relva;
 - f) Fazer crescer a relva de tempos a tempos;
 - g) Construir e atualizar o gráfico.

3. Exercício – Criação de um modelo básico

De forma a consolidar os conhecimentos, seguir as tarefas seguintes:

- Criar um novo ficheiro (File → New) e grave-o na sua conta (File → Save);
- Alterar propriedades da grelha bidimensional para que esta tenha *max-pxcor* igual a 12, *max-pycor* igual a 12 e *patch size* igual a 15;
- No separador *Interface*, crie o botão **Setup** e o botão **Go** (botão **Add** ao lado do tipo de botão a criar), ativando a opção *forever* para o botão **Go**, e associando a cada um dos botões os comandos (ou procedimentos) com o respetivo nome, conforme se exemplifica na figura 11;

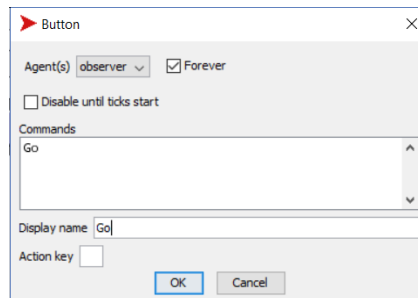


Figura 11 – Alteração das características de um botão.

- No separador *Code*, escrever o código dos procedimentos associados a cada um dos botões:

<pre> to Setup clear-all create-turtles 10 ask turtles [set shape "person" set color yellow] end </pre>	<pre> to Go ask turtles [forward 1] end </pre>
Cria 10 pessoas, todas no mesmo sítio	Cada pessoa avança 1 unidade

- Ir para o separador *Interface* e executar a simulação do modelo, pressionando primeiro no botão **Setup** (que inicializa o modelo) e depois no **Go**;
- Alterar a posição e a orientação inicial das pessoas (*turtles*), acrescentando o código a vermelho no procedimento **Setup**:

```
to Setup
  clear-all
  create-turtles pessoas
  ask turtles
  [
    setxy random-xcor random-ycor
    set heading 90
    set shape "person"
    set color yellow
  ]
end
```

- Executar novamente a simulação do modelo e ver o resultado desta mudança de código (pode diminuir a velocidade de visualização no *slider* da barra principal para ver melhor os efeitos do código introduzido – figura 12);

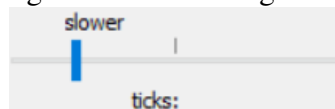


Figura 12 – Botão para reduzir/aumentar a velocidade de movimento dos gráficos do modelo.

- No código do procedimento **Setup**, alterar o parâmetro de orientação de movimento, o **heading**, de 90 para 0 e ver a diferença ao executar a simulação do modelo;
- Colocar as pessoas num movimento aleatório, alterando a sua orientação (**heading**) para **random 360**;
- Acrescentar um **Slider** para controlar o número de pessoas. Para isso:
 - a. Na janela que aparece aquando da criação do **Slider**, definir uma variável global, de nome **pessoas**, que tenha possa ter valores entre 0 e 10 e com o valor inicial de 5, conforme a figura 13;

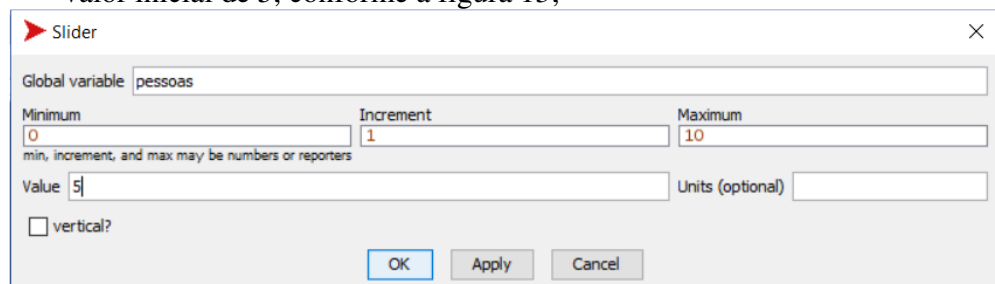


Figura 13 – Alteração das características do **Slider**.

- b. Alterar no código do procedimento **Setup** de maneira a que o número de **turtles** a criar corresponda ao valor da variável **pessoas**, inicializada no *slider* que introduziu anteriormente, escrevendo:

```
create-turtles pessoas
```

- A energia de uma *turtle* normalmente está ligada às tarefas que faz (por exemplo, perde energia ao se movimentar e ganha energia ao se alimentar). Assim, acrescentar energia às pessoas, colocando a seguinte linha de código no topo do ficheiro:

```
turtles-own [energia]
```


- Inicializar cada pessoa com um valor de energia aleatória positiva, de valor máximo 100, acrescentando o código a vermelho no procedimento **Setup**:

```
to Setup
  ...
  ask turtles
  [
    set energia random 101
    ...
  ]
end
```

- Colocar energia de cada pessoa a diminuir uma unidade a cada movimento que fazem, a pessoa a morrer quando a sua energia for menor ou igual a zero e marcar a vermelho o local onde as pessoas morreram, acrescentando o código a vermelho no procedimento **Go**:

```
to Go
  ask turtles
  [
    ...
    set energia energia - 1
    if energia <= 0
    [
      ask patch-here
      [
        set pcolor red
      ]
      die
    ]
  ]
end
```

- Para que a simulação do modelo pare quando não existirem pessoas (para o botão *forever*), acrescentar o código a vermelho no procedimento **Go**:

```
to Go
  ask turtles
  [
    ...
  ]
  if (count turtles = 0)
  [
    stop
  ]
end
```

- Executar a simulação do modelo e verificar todas as alterações pedidas e apresentadas acima;
- Para que a simulação do modelo tenha em conta a alimentação das pessoas (este alimento será representado por *patches* verdes e dará energia às pessoas que chegam a essas *patches*. Essa energia serve para se conseguirem movimentar durante mais tempo), fazer o seguinte:

- a. Acrescentar o código a vermelho no procedimento **Setup** (código que manda executar um novo procedimento chamado **Setup-patches**):

```
to Setup
  clear-all
  Setup-patches
  create-turtles pessoas
  ask turtles
  [
    ...
  ]
end
```

- b. Acrescentar o procedimento **Setup-patches** ao código já existente, o qual coloca, de forma aleatória, algumas *patches* a cor verde, na área de simulação:

```
to Setup-patches
  ask patches
  [
    let x random 101 ; número aleatório entre 0 e 100
    if x < 10
    [
      set pcolor green
    ]
  ]
end
```

Isto é um comentário
(coloca-se um ; antes
do comentário)

- c. Acrescentar o código a vermelho no procedimento **Go** (código que manda executar um novo procedimento chamado **Comer**):

```
to Go
  ask turtles
  [
    ...
    Comer
  ]
  ...
end
```

- d. Acrescentar o procedimento **Comer** ao código já existente, o qual deteta a cor da *patch* e, se esta for verde, passa-a a preto e aumenta a energia da pessoa:

```
to Comer
  if pcolor = green
  [
    set pcolor black
    set energia (energia + 100)
  ]
end
```

- Executar a simulação do modelo e verificar o seu resultado com estas novas alterações;
- Adicionar um **Monitor** para ver o número de pessoas que se encontram vivas ao longo da execução da simulação do modelo e na janela que aparece após essa ação escreva o que se encontra na figura 14;

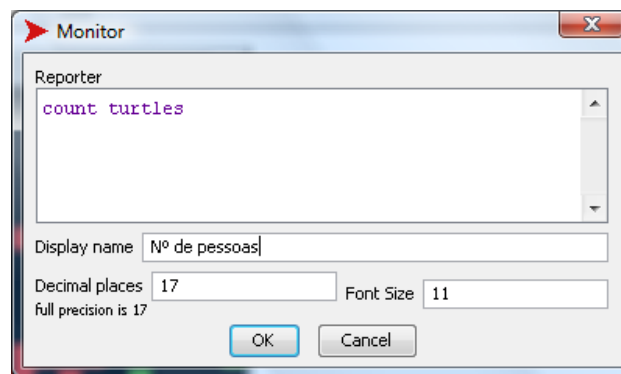


Figura 14 – Alteração das características do *Monitor*.

- A informação do monitor pode ser vista em forma de gráfico. Para se fazer isso,
 - a. Acrescentar um *Plot* e na janela que aparece após essa ação coloque o que se encontra na figura 15;

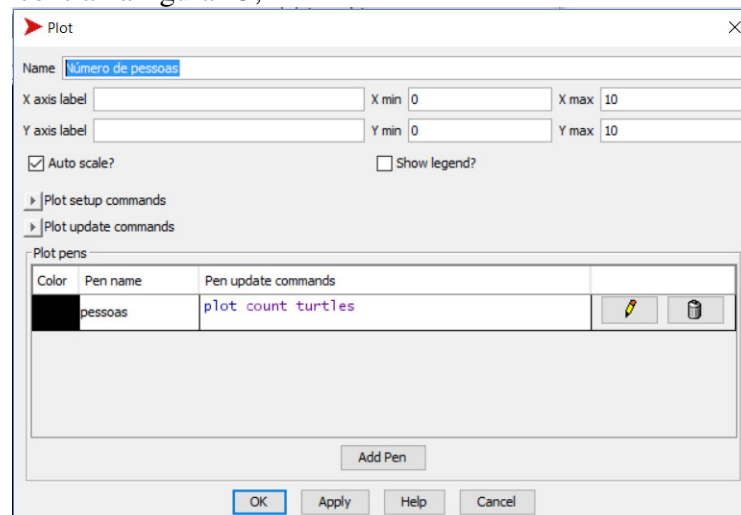


Figura 15 – Alteração das características do *Plot*.

- b. A atualização do gráfico ao longo do tempo é feita através das seguintes ações:

- i. No procedimento em *Go*, acrescentar a instrução *tick*:

```
to Go
...
  if (count turtles = 0)
    [stop]
  tick
end
```

- ii. No procedimento em *Setup*, adicionar a instrução *reset-ticks*:

```
to Setup
...
  reset-ticks
end
```

- A instrução *breed* permite criar vários tipos de *turtles*. Assim, para que o modelo criado possa ter dois tipos de pessoas, homens e mulheres,
 - a. Colocar as seguintes linhas de código no topo do ficheiro:

```
breed [homens homem]
breed [mulheres mulher]
```

- b. Comentar a linha *create-turtles pessoas* e *set color yellow*, no procedimento *Setup*;
- c. Editar o *Slider* que mostra o número de pessoas e alterar a variável global para *nmulheres*;
- d. Criar mais um *Slider* no interface gráfico, similar ao anterior, para definir o número de mulheres, com a variável global *nhomens*;
- e. No procedimento em *Setup*, adicionar o código a vermelho, o qual vai atribuir às mulheres a cor vermelha e tamanho 1.5 e dar aos homens cor azul e tamanho 1:

```
to Setup
...
; create-turtles pessoas
create-mulheres nmulheres
create-homens nhomens
ask mulheres
[
  set size 1.5
  set color red
]
ask homens
[
  set size 1
  set color blue
]
ask turtles
[
  ...
  ; set color yellow
  ; Aqui ficam apenas as instruções comum aos dois tipos
]
end
```

- f. Para que as mulheres e os homens gastem, respetivamente, 2 e 1 unidades de energia, quando se movimentarem, e que, quando morrerem, as mulheres deixem uma *patch* vermelha e os homens uma azul, altere o procedimento *Go*, acrescentando o seguinte código a vermelho:

```
to Go
ask turtles
[
  forward 1
  ifelse breed = mulheres
  [
    set energia (energia - 2)
  ]
  [
    set energia (energia - 1)
  ]
]
```

```
if energia <= 0
[
  ifelse breed = mulheres
  [
    set pcolor red
  ]
  [
    set pcolor blue
  ]
  die
]
comer
]
...
end
```

- Editar o gráfico e alterá-lo para que tenha duas canetas, uma para registrar o número de homens e outra para registrar o número de mulheres, conforme se pode verificar na figura 16.

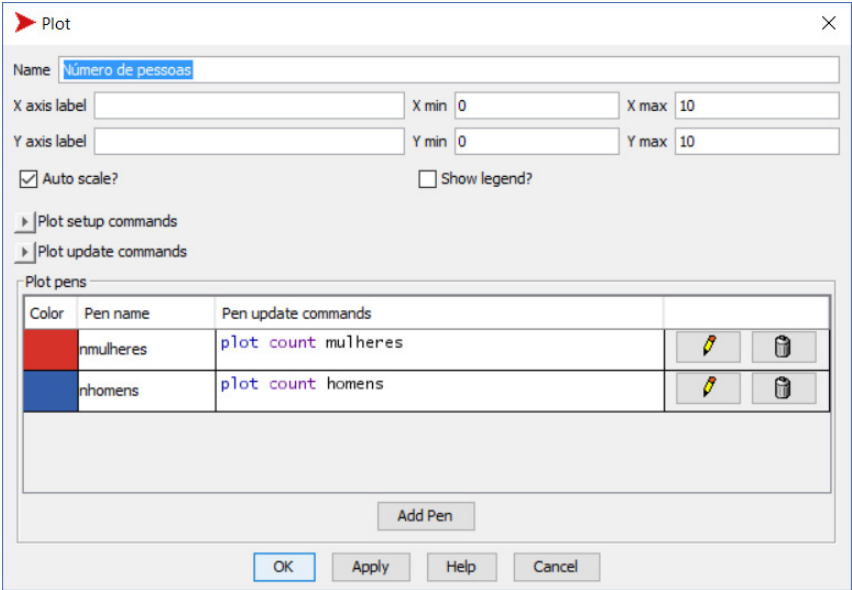


Figura 16 – Alteração das características do *Plot*, para mostrar graficamente o número de mulheres e de homens separadamente.