به نام پرتو بخش هستی

فهرست محتوا

سخنی با پارسی زبانان عزیز 3
فصل اول: مقدمه 5
NetLogo چیست؟ 5
مشخصه ها 6
یک مدل نمونه: مهمانی 8
در مهمانی 8
چالش فکری 12
تفکر به کمک مدل سازی 12
آنچه در ادامه خواهد آمد 13
فصل دوم: یادگیری NetLogo 14
آموزش شماره 1: مدل ها 14
مدل نمونه: شکار گوسفند گرگ 15
کنترل مدل: دکمه ها 16
کنترل سرعت: اسلایدرها و سوییچ ها 18
گردآوری اطلاعات: نمودارها و مانیتورها 21
نمودارها 21
مانیتورها 22
کنترل کردن نمای مدل 22
کتابخانه مدلها 27
مدلهای نمونه 27
مدل های برنامه درسی 27
مثالهای کد 28
فعالیت های HubNet 28
آنچه در ادامه خواهد آمد 28
آموزش شماره 2: فرمانها 29
مدل نمونه : مدل پایه ترافیک 29
مرکز فرمان 30

کارکردن با رنگها 33
مانیتورهای عامل و فرماندههای عامل 36
آنچه در ادامه خواهدآمد 40
آموزش شماره3: رویه ها 41
عاملها و رویه ها 42
ایجاد دکمه setup 42
برگشت به به روزرسانی های مبتنی بر تیک 46
ایجاد دکمه go 46
آزمایش با فرمانها 49
پَچ ها و متغیرها 50
متغیرهای تِرتِل 51
مانیتورها 54
سوییچ ها و برچسب ها 56
رویه های بیشتر 58
ترسیم نمودار 60
شمارنده تیک 62
برخی جزییات بیشتر 63
آنچه در ادامه خواهد آمد 64
ضمیمه: کد کامل 66

سخنی با پارسی زبانان عزیز

ما در جهان پویایی زندگی میکنیم؛ جایی که هر روز شاهد پدیدههایی با سطوح مختلفی از پیچیدگی هستیم. از سرایت ویروس در محیط و انتشار شایعه در جوامع انسانی گرفته تا شکلگیری هنجارهای اجتماعی مختلف، ظهور فناوریهای جدید و نوآوریهای برافکن همه و همه نمودی از پدیدههای جهان ماست. به منظور مطالعه و شناخت بهتر این پدیدهها، دانشمندان علوم اجتماعی در بسیاری از مواقع از رویکرد تقلیلگرایی[1] استفاده می کنند. جایی که آنها چنین پدیدههایی را به متغیرهای سطح پایینتری تقسیم و تقلیل

[1] Reductionism

میدهند و با بکارگیری بدنهای از معادلات[2] (به عنوان نمونه، معادلات دیفرانسیل جزیی و معمولی)، روابط بین این متغیرها را مدلسازی میکنند. رویکرد تقلیلگرایی که گاه از آن بعنوان مدلسازی معادله محور[3] (EBM) نیز نامیده میشود در پرداختن به مسایل پیچیدهی دنیای واقع از محدودیتهای بنیادینی برخوردار است. به عنوان نمونه در مدلسازی نحوهی شکلگیری یک تحول اجتماعی، کاربران این رویکرد، ابتدا سیستم مورد مطالعه را به عاملهایی (مثلاً عاملهای اقتصادی، نهادی و ...) با عقلانیت نامحدود و اطلاعات غالباً کامل تقسیم میکنند و با ایجاد روابطی در بین این عاملها اقدام به توسعهی مدلی برای توضیح چگونگی این تحول اجتماعی مینمایند، حال آنکه موارد بسیار مهمی نظیر بستر تاریخی، انطباقپذیری بازیگران و ماهیت تطوری تمامی آنها به همراه نقش و اهمیت اثرات شبکهای محیطی که بازیگران در آن زندگی میکنند را نادیده میگیرند. در راستای رفع نواقص رویکرد تقیلگرایی، در خلال دو دههی گذشته، رویکرد سیستمهای وفقی پیچیده[4](CAS) کاربردهای عظیمی یافته است. برخلاف رویکرد تقیلگرایی، در این رویکرد، پدیدههای اقتصادی اجتماعی نظیر تحولات اجتماعی و غیره به نحوی ارگانیک و پویا مطالعه میشوند، جاییکه که عاملها (بازیگران) نه تنها از عقلانیتی محدود برخوردارند که ماهیتی قویاً انطباقی و یادگیرنده نیز دارند و در تعاملی دو سویه با شبکه اجتماعی محیط پیرامون خود نیز میباشند. بر اساس این رویکرد پدیدههای اقتصادی- اجتماعی یک سیستم در واقع برآیندی (نوخاستهای)[5] از شیوهی تعاملات بازیگران آن سیستم و منش تصمیمگیری آنها است و نمیتوان آن را به تک تک اجزای جامعه تقلیل داد. به عنوان برجستهترین روش مدلسازی سیستمهای وفقی پیچیده، مدلسازی عامل محور[6] (ABM)در بین اندیشمندان معاصر به محبوبیت زیادی رسیده است. این روششناسی نشان میدهد که چگونه قواعد رفتاری وگاه سادهی عاملها و تعاملات محلی میان آنها در سطح خرد میتواند الگوهای بسیار پیچیدهای را در سطح کلان (جامعه) ایجاد کند.

برای پیادهسازی مدلهای عامل محور ابزارهای متعددی طراحی شده اند که یکی از پرکاربردترین آنها Netlogo میباشد. یادگیری این ابزار میتواند برای هموطنانی که علاقمند به یادگیری ABM هستند بسیار مفید و کاربردی باشد که هدف اصلی این راهنما نیز فراهم کردن چنین دانشی برای پارسیزبانان عزیز است. بنابراین، سعی شده تا بخشهای کاربردی از کتابچه راهنمای NetLogo ترجمه شود. بعنوان جامعترین منبع برای تمامی افرادی که به یادگیری NetLogo علاقمندند، این کتابچه راهنما همگام با توسعه NetLogo نیز به روزآوری میشود. عزیزان میتوانند با مراجعه به https://ccl.northwestern.edu/netlogo/docs/ نسخه به روز شده این کتابچه را دریافت و مطالعه نمایند.

در نوشتار حاضر تنها ترجمهی قسمتهای اول و دوم صورت پذیرفته است. چون باور بر این است که فهم این دو قسمت تا حد زیادی برای کارکردن با NetLogo کفایت میکند و افزون براین خوانندگان عزیز را

[2] Scheme of Equations
[3] Equation-based Modeling
[4] Complex Adaptive Systems
[5] Emergence
[6] Computational Agent based Modeling

برای یادگیری مطالب پیشرفتهتر نیز آماده میسازد. در این نوشتار هر قسمت را بعنوان یک فصل نامیدهایم. قسمت اول که"مقدمه"نام دارد به دو بخش تعاریف و مشخصههای نرمافزاری NetLogo تقسیم میشود. قسمت دوم که "یادگیری NetLogo نام دارد" از سه بخش اساسی تشکیل شده است که هر کدام حکم یک دورهی آموزشی را دارد. بخش اول به آموزش چگونگی استفاده از مدلهای NetLogo پرداخته است. بخش دوم به آموزش چگونگی دستکاری و تغییر مدلهای NetLogo تخصیص یافته است و در نهایت در بخش سوم به آموزش چگونگی ساخت یک مدل جدید در NetLogo به صورتی گام به گام پرداخته شده است. افزون بر این، به منظور درک بهتر مطالب، پانوشتهایی در صفحات آمده است که به خوانندگان عزیز توصیه میشود حتماً آنها را نیز مطالعه نمایند.

در انتها، ترجمه حاضر را به استاد برجستهی علم و اخلاق جناب **دکتر محمد علی شفیعا** تقدیم میکنم و برای تمامی همزبانان عزیزم آرزوی سلامتی و توفیقات روزافزون دارم و امیدوارم در پرتو رحمت الهی همراه ارزش آفرین باشند.

ارادتمند تمامی عزیزان
حسین سبزیان
sabzeyan@yahoo.com
تهران مهرماه 1399

فصل اول: مقدمه

این فصل در دو بخش تدوین شده است. بخش نخست به تعریف NetLogo پرداخته است. بخش دوم نیز مشخصههای فنی آن را تشریح کرده است[7].

NetLogo چیست؟

NetLogo یک محیط مدلسازی قابل برنامهنویسی برای شبیهسازی پدیدههای طبیعی و اجتماعی است. NetLogo در سال 1999 توسط یوری وایلنسکی[8] نوشته شد و از آن زمان تاکنون به نحوی پیوسته توسط مرکز یادگیری متصل و مدلسازی کامپیوتری (CCL)[9] دانشگاه نورثوسترن توسعه یافته است.

[7] برای دانلود NetLogo و آگاهی از الزامات نصب آن رجوع شود به https://ccl.northwestern.edu/netlogo/download.shtml (مترجم).

[8] Uri Wilensky

[9] Center for Connected Learning and Computer-based Modeling

NetLogo برای مدلسازی سیستمهای پیچیدهای که دارای ماهیت تکاملی میباشند بسیار مناسب است. مدلسازان میتوانند به صدها هزار عامل مستقل از هم دستورالعمل بدهند. این باعث میشود تا بتوان رابطهی بین رفتار اجزای سیستم در سطح خرد و رفتاری که در سطح سیستم پدیدار میشود را تحلیل کرد.

NetLogo به دانش آموزان اجازه میدهد تا برنامههای شبیهسازی را باز کرده، با آنها بازی کنند و رفتار آنها را در شرایط مختلف بررسیکنند. این نرم افزار همچنین یک محیط نگارشی است که دانشآموزان، معلمان و توسعهدهندگان محتوای درسی را قادر میسازد تا مدلهای ذهنی خود را در آن بنویسند. یادگیری NetLogo برای دانشآموزان و معلمان به اندازه کافی ساده است، در عین حال، ابزار بسیار قدرتمندی است که در طیف گستردهای از حوزههای پژوهشی میتوان از آن استفاده کرد.

NetLogo مستندات و آموزشهای گستردهای دارد که با کتابخانهای از مدلها و مجموعهی گستردهای از شبیهسازیهای از پیش نوشته شده که امکان استفاده و تغییر آنها وجود دارد، پشتیبانی میشود. این شبیهسازیها زمینههای مختلفی از علوم طبیعی و اجتماعی مانند زیستشناسی و پزشکی، شیمی و فیزیک، ریاضی و علوم کامپیوتر و اقتصاد و روانشناسی اجتماعی را در برمیگیرند. افرون بر این، برنامههای درسی متعددی در خصوص " تحقیق مبتنی بر مدل[10]" وجود دارد که توسط NetLogo نوشته شده اند و بسیاری هم در حال توسعه میباشند.

NetLogo نسل بعدی از سری زبانهای مدلسازی چندعامله نظیر Starlogo و StarlogoT میباشد. این زبان روی ماشین مجازی جاوا (JVM) اجرا میشود از اینرو در تمام سیستم عاملهای عمده (Mac, Windows, Linux) کار میکند. این نرم افزار بعنوان یک برنامه دسکتاپ اجرا میشود[11] و از عملیات خط فرمان نیز پشتیبانی[12] میکند.

مشخصه ها

- سیستم:
  - رایگان و منبع باز
  - چندسکویی[13]، قابلیت اجرا روی linux، Windows، Mac و غیره
  - پشتیبانی از زبانهای غیربومی [14]
- برنامهنویسی:
  - کاملاً قابلبرنامهنویسی

[10] Model-based inquiry

[11] یکی از کوششهای جدی که به تازگی آغاز شده است. توسعه NetLogo موبایل است که امکان پیاده سازی NetLogo بر بستر دستگاههای موبایل را فراهم میسازد برای اطلاع بیشتر در این زمینه رجوع شود به https://ccl.northwestern.edu/2020/Final%20Draft.pdf (مترجم).

[12] Command Line Operation

[13] Cross-platform

[14] International character set support

- سینتَکس قابل فهم[15]
- گویشی از زبان Logo است که برای پشتیبانی از عاملها توسعه یافته است.
- عاملهای متحرک (تِرتِلها[16]) میتوانند روی محیطی از عاملهای ثابت (پَچها[17]) حرکت کنند.
- عاملهای ارتباطی (لینکها[18]) میتوانند تِرتِلها را به هم وصل نموده و شبکههای مختلفی ایجاد کنند.
- فهرست عظیمی از سینتَکسهای از پیش تعریف شده
- فرمت نقطه شناور دقت دوگانه[19] (نمایش رقمی طولانی از یک عدد اعشاری)
- مقادیر توابع کلاس اول (نظیر رویههای بینام، کلوژر و لامبدا)
- برنامههای نوشته شده با این زبان روی پلتفرمهای مختلف قابل اجراست.
- قابلیت اجرا در روی سختافزارهای مختلف

- محیط:
  - مرکز فرمان برای تعاملات فوری[20]
  - داشتن عناصر واسط متنوع نظیر دکمه، اسلایدر، سوییچ، انتخابکننده[21]، مانتیور، کادر متنی، خروجی و نمودار
  - داشتن تب اطلاعات برای مستند سازی مدل از طریق تصاویر و متن فرمت شده
  - ارایه فناوری هابنت[22] برای شبیهسازی مشارکتی با بهرهگیری از ابزارهای شبکه
  - مانتیورهای عامل برای بررسی و کنترل عاملها
  - توابع وردی و خروجی (استخراج داده، ذخیرهسازی و بازیابی حالتی از مدل و ایجاد فیلم)
  - ارایه فناوری behaviorSpace بعنوان یک ابزاری منبع باز که برای جمع آوری داده ها از چندین اجرای موازی یک مدل استفاده می شود.
  - امکان مدلسازی System Dynamics
  - نتلوگوی سه بعدی (3D) برای مدلسازی محیط سه بعدی
  - امکان اجرای مدل به صورت batch[23]
- نمایش و بصریسازی
  - نمودارهای خطی، میلهای و پراکنده
  - اسلایدر سرعت به شما امکان را میدهد که سرعت مدل را کاهش یا افزایش بدهید.

---

[15] در NetLogo از سینتَکسها بنام Primitive یاد میشود که به چند نوع فرمان، ریپورتر، مقادیر ثابت عددی و کلمات کلیدی تقسیم میشوند (مترجم).
[16] Turtles
[17] Patches
[18] Links
[19] Double precision floating point math
[20] On-the-fly interaction
[21] Chooser
[22] HubNet
[23] به این معنی که شما یک برنامهای خارج از NetLogo بنویسید و از آن طرف مدل را اجرا کنید از این نوع اجرا بعنوان batch running یاد می شود (مترجم).

- امکان مشاهده مدل در یک محیط دو بعدی یا سه بعدی
- اشکال بردای قابل چرخش و مقیاسپذیر
- برچسبهای تِرتِل و پَچ.

- API ها:
  - API کنترلگر[24] به شما این امکان را میدهد تا NetLogo را در داخل یک اسکریپت یا برنامه قرار دهید.
  - بسطهایAPI[25] امکان افزودن فرمانها و ریپورترهای[26] جدیدی را به زبان NetLogo فراهم می کند. مثالهای از این بسطها در خود NetLogo وجود دارد.

یک مدل نمونه: مهمانی

این فعالیت باعث میشود تا شما دربارهی مدلسازی کامپیوتری و چگونگی استفاده از آن بیاندیشید. این فعالیت شما را با NetLogo نیز آشنا میسازد. بنابراین ما به کاربران تازهکار پیشنهاد میدهیم از اینجا شروع کنند.

- **در مهمانی**
- **چالش**
- **تفکر به مدد مدلسازی**
- **آنچه در ادامه خواهد دید**

در مهمانی

آیا تاکنون در مهمانی به این اندیشیدهاید که چگونه افراد در گروهای مختلف دستهبندی میشوند؟ شما همچنین شاید متوجه شدهاید که افراد به ندرت در یک گروه میمانند و دایم در حال چرخ زدن هستند که با این کار اندازه گروهها دایم تغییر میکند. اگر شما در طول زمان به این تغییرات نگاه کنید، متوجه یکسری الگوها خواهید شد.

به عنوان مثال، در محیطهای اجتماعی نظیر مهمانی ، افراد ممکن است رفتارهای متفاوتی نسبت به محل کار یا خانه نشان دهند. افرادی که در محیط کار خود پرجنب و جوش هستند ممکن است در یک گردهمآیی[27] اجتماعی خجالتی و ترسو شوند. و دیگران که در محل کار محفوظ هستند ممکن است در چنین گردهمآییهایی بسیار پرجنب و جوش باشند.

[24] Controlling API
[25] API Extensions
[26] reporter
[27] Gathering

این الگوها میتوانند به نوع گردهمآیی (افراد) وابسته باشند. در برخی از محیطها، افراد تمایل دارند تا در گروههای مختلط خود را ساماندهی کنند. بعنوان نمونه، در مهمانی بازی[28] یا فعالیتهای مربوط به مدرسه، اما در محیطهای غیرساختاریافته، افراد فعالیتهای گروهسازی را به نحوی تصادفیتر انجام میدهند.

آیا این گروهبندی دارای نوعی از الگو است؟

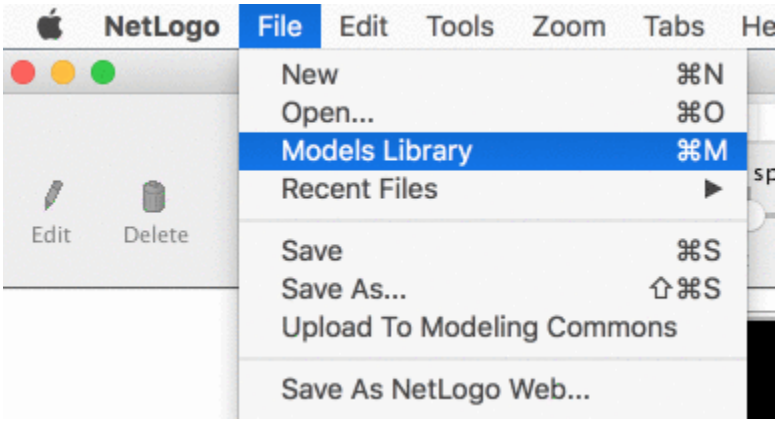
بیایید با شبیهسازی کامپیوتری رفتار افراد در مهمانی، نگاهی عمیقتر به این موضوع داشته باشیم. مدل مهمانی[29] در نرم افزار NetLogo مشخصا به موضوع گروهبندی بر اساس جنسیت[30] در مهمانی پرداخته است.

حال بیایید از NetLogo برای بررسی عمیقتر این سوال استفاده کنیم.

مراحل کار

گام اول: نرم افزار NetLogo را باز کنید.

گام دوم: از منوی File گزینه Models Library را بازکنید.



گام سوم: پوشه Social science را انتخاب کنید.

گام چهارم: روی مدلی که نام آن Party است کلیک کنید.

گام پنجم : دکمه Open را فشار دهید.

گام ششم: دکمه setup را فشار دهید.

در نمای مدل[31] مدل، شما گروههای آبی و صورتی شمارهداری را مشاهده میکنید.

[28] Party game
[29] Party model
[30] Grouping by gender
[31] View

این خطوط نمایانگر گروه های درهم و برهم در یک مهمانی هستند. مردان به رنگ آبی ، زنان صورتی نشان داده می شوند. اعداد اندازه گروهها هستند.

آیا تمام گروهها تقریباً تعداد اعضای مشابه دارند؟

آیا جنسیت افراد تمام گروها تقریباً مشابه است؟

فرض کنید شما یک مهمانی گرفتهاید و 150 نفر را دعوت کردهاید. شما تعجب میکنید که چگونه مردم دور هم جمع میشوند. فرض کنید 10 گروه در مهمانی تشکیل شده است.

فکر میکنید که آنها چگونه گروهبندی می شوند؟

بجای اینکه از 150 نفر از نزدیک ترین دوستانتان دعوت کنید تا دورهم جمع شوند و به صورت تصادفی گروههایی ایجاد کنند بیایید با کامپیوتر این وضعیت را شبیهسازی کنیم.

مراحل کار:
گام اول: دکمه go را فشار دهید[32]. (فشار مجدد دکمه go به صورت دستی باعث توقف مدل میشود)
گام دوم: به حرکت افراد توجه کنید تا مدل متوقف شود.
گام سوم: به نمودارها هم نگاه کنید و ببینید چه اتفاقی دارد می افتد.
گام چهارم: اگر لازم است سرعت مدل را کم کنید از اسلایدر سرعت استفاده کنید.

حال چند نفر در هر گروه قرار دارد؟

در ابتدا، شما شاید فکر میکردید که 150 نفری که در 10 گروه قابل تقسیم اند در نتیجه چنین خواهد شد که هر گروه تقریبا 15 نفر خواهد داشت. ولی مدل نشان می دهد که افراد به طور مساوی در 10 گروه تقسیم نمیشوند بطوریکه برخی گروهها بسیار کوچک اند و برخی نیز بسیار بزرگ اند. افزون بر این، گروههای مختلط در گذر زمان به گروههای تک جنسی تغییر یافتند.

چه چیزی میتواند این را توضیح دهد؟

پاسخهای احتمالی زیادی میتوان به این سوال داد. طراح این شبیهسازی فکر کرد که گروههای مهمانی فقط به طور تصادفی تشکیل نمی شوند. گروهها با نحوه رفتار افراد در مهمانی مشخص می شوند. طراح تصمیم گرفت تا روی متغیر خاص، "tolerance" متمرکز شود:

[32] توجه داشته باشید شاید تعداد افراد به صورت پیش فرض روی 150 نفر تنظیم نشده باشد که شما به راحتی میتوانید از اسلایدر تعداد، ان را تغییر دهید و دوباره مدل را setup کنید (مترجم).

متغیر tolerance به سطحی اشاره دارد که تا آن حد فرد میتواند افرادی با جنس مخالف را در گروه خود تحمل کنند. بعنوان نمونه، اگر tolerance فردی 25% باشد او فقط درگروهایی عضو میشود که حداکثر 25% جنسیت افراد آنها با او متفاوت باشد و اگر این تفاوت بیشتر شود او احساس راحتی نمیکند از گروه خارج میشود و بدنبال گروه دیگری میرود. وقتی افراد ناراحت میشوند و گروهها را ترک میکنند، به گروههای جدیدی می پیوندند که ممکن است باعث شود بعضی افراد در آن گروه به نوبه خود "نارحت" شوند. این واکنش زنجیره ای ادامه می یابد تا همه افراد در این مهمانی در گروهی قرار گیرند که در آن احساس راحتی کنند.

توجه کنید که tolerance یک متغیر است و مقدار آن ثابت نیست. شما، بعنوان، کاربر می توانید از اسلایدر tolerance برای تغییر مقدار آن استفاده کنید و با شروع مجدد مدل، نتیجه را مشاهده کنید.

چگونه دوباره شروع کنید:
گام اول: اگر دکمه "go"را فشار بدهید (رنگ مشکی). سپس مدل مجدد اجرا میشود. فشار مجدد دکمه باعث میشود مدل متوقف شود.
گام دوم: با کشیدن دسته قرمز اسلایدر "tolerance"، مقدار آن را تغییر دهید.
گام سوم: برای تنظیم مجدد مدل ، دکمه "setup" را فشار دهید.
گام چهارم: دکمه "go" را فشار دهید تا مدل دوباره اجرا شود

چالش فکری

شما به عنوان میزبان مهمانی ، دوست دارید هر گروهی ترکیبی از زن و مرد باشد. حال اسلایدر tolerance را طوری تغییر دهید که در نهایت تمام گروها مختلط گردند.

مطمئن شوید که تمام 10 گروه از هر دو جنس مخلف برخوردارند، پارامتر tolerance را در چه سطحی باید قرار داد؟

پیشبینیهای خودتان را روی مدل تست کنید.

آیا می توانید عوامل یا متغیرهای دیگری را مشاهده کنید که ممکن است در هر گروه نسبت زن و مرد را تحت تأثیر قرار دهد؟

پیشبینیهای خود را انجام داده و ایده های خود را درون این مدل آزمایش کنید.

همانطور که شما دارید فرضیاتتان را آزمون میکنید، شما متوجه خواهید شد که الگوهای جدیدی از این داده (تغییر اسلایدر) ظاهر میشود. بعنوان مثال اگر تعداد افراد را در میهمانی ثابت نگه دارید اما به تدریج سطح tolerance را افزایش دهید، گروههای مختلط بیشتری ظاهر می شوند.

قبل از اینکه گروههای مختلطی ایجاد کنید، سطح tolerance چقدر باید بالا باشد؟

چه درصدی از tolerance باعث ایجاد چه درصدی از اختلاط دارد؟

تفکر به کمک مدل سازی

استفاده از NetLogo برای مدلسازی وضعیتی مانند یک مهمانی به شما این امکان را می دهد تا یک سیستم را به روشی سریع و انعطافپذیر آزمایش کنید. چیزی که انجام آن در دنیای واقعی دشوار خواهد بود. مدلسازی همچنین این امکان را به شما می دهد که یک وضعیت یا شرایط را با پیش فرض های کمتری مشاهده کنید، زیرا می توانید پویایی اساسی یک موقعیت را بررسی کنید. شما ممکن است متوجه شوید که هرچه بیشتر و بیشتر مدلسازی را انجام میدهید، بسیاری از ایده های از پیش تصور شده شما در مورد پدیده های مختلف به چالش کشیده میشوند. به عنوان مثال، یک نتیجه شگفت آور از مدل Party این است که حتی اگر سطح tolerance نسبتاً بالا باشد ،(باز هم) جدایی زیادی بین جنسهای مخالف ایجاد میشود.

این یک مثال کلاسیک از یک پدیده نوخاسته (نوظهور)[33] میباشد جایی که در آن یک الگویی از گروه در نتیجه تعاملات افراد متعددی ایجاد میشود. چنین ایدهای (از پدیده نوخاسته) را می توان تقریباً به هر موضوعی تعمیم داد.

در مورد کدام پدیده های نوخاسته دیگر می توانید فکر کنید؟

برای دیدن مثالهای بیشتر و درک عمیقتری از این مفهوم و اینکه چگونه NetLogo می تواند یادگیرندگان را قادر سازد تا بیشتر این مفهوم را بررسی کنند، شما را تشویق می کنیم که کتابخانه مدلهای NetLogo[34] را بررسی کنید. این کتابخانه شامل مدلهایی است که این ایده (یعنی پدیدههای نوخاسته) را در سیستم های مختلف نشان می دهد[35]. برای مطالعه عمیق تر مفهوم نوخاستگی و کارکردهای نرم افزار NetLogo در شناخت این موضوع، مقاله /http://ccl.northwestern.edu/papers/MEE به خوانندگان گرامی توصیه میشود.

آنچه در ادامه خواهد آمد

بخشی از راهنمای کاربری بنام **آموزش شماره 1: مدلها** با جزییات بیشتری به چگونگی استفاده از سایر مدلها در کتابخانه مدلها میپردازد.

[33] Emergent phenomenon
[34] Netlogo's Models Library
[35]

اگر میخواهید عمیقتر مدلها را بررسی کنید، **آموزش شماره2: فرمانها** زبان برنامه نویسی NetLogo را به شما معرفی خواهد کرد.

در نهایت، شما برای **آموزش شماره 3: رویهها** آماده خواهید شد. جایی که شما یاد میگیرید که چگونه مدلهای موجود را بسط و تغییر داده و رفتارهای جدیدی را ایجاد کنید و شما می توانید شروع به ساخت مدل خودتان بنمایید.

فصل دوم: یادگیری NetLogo

این فصل از سه بخش تشکیل شده است. در بخش اول که به آموزش شماره 1 اختصاص دارد، شما یاد میگیرید که چگونه از مدلهای Netlogo استفاده کنید. در بخش دوم که به آموزش شماره 2 اختصاص دارد به شما آموزش داده خواهد شد که چگونه مدلهای Netlogo را دستکاری کنید و در نهایت در بخش سوم که در برگیرنده آموزش شماره 3 است شما خواهید آموخت که چگونه مدل مد نظر خودتان را در Netlogo طراحیکنید.

آموزش شماره 1: مدل ها

اگر شما بخش **یک مدل نمونه: مهمانی** را خوانده باشید، شما یک اطلاعات کلی درخصوص چگونگی کارکردن و کاوش در مدلهای NetLogo بدست آوردهاید. این بخش در سطح عمیقتری به این موضوع میپردازد و مشخصههای که در حین کار با مدلها با آنها روبرو میشوید را بیشتر تحلیل میکند.

در خلال تمام این آموزش، ما همواره از شما خواهیم پرسید که پیامدهای تغییراتی که شما در مدل ایجاد کردهاید را پیشبینی کنید. به خاطر داشته باشید که این پیامدها اغلب غیرمنتظره (و تعجب آور) میباشند. ما بر این باوریم که ماهیت غیرمنتظره بودن این پیامدها بسیار هیجان انگیز است و فرصتهای خاصی برای یادگیری فراهم میسازد.

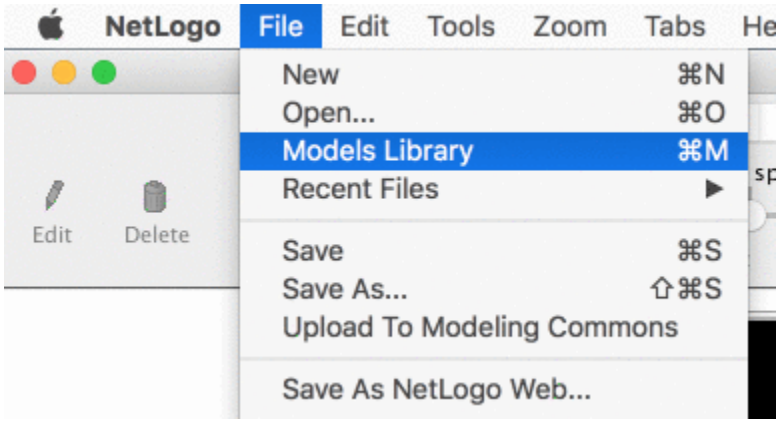
ممکن است شما بخواهید در حالیکه از NetLogo استفاده میکنید، آموزشهای خود را هم پرینت بگیرید تا مراجعه به آنها راحتتر شود.

- مدل نمونه: شکار گوسفند گرگ
- کنترل مدل: دکمهها
- کنترل سرعت: اسلایدرها و سوییچها
- گردآوری اطلاعات: نمودارها و مانیتورها
- کنترل کردن نمای مدل
- کتابخانه مدلها
- آنچه در ادامه خواهد آمد

مدل نمونه: شکار گوسفند گرگ

ما یکی از مدلهای نمونه را باز خواهیم کرد و با جزییات بیشتری آن را بررسی خواهیم نمود. بیایید یک مدل زیست شناسی[36] را بررسی کنیم که نام آن مدل شکار گوسفند گرگ[37] می باشد.

- از منوی File گزینه Models Library را بازکنید.



- مدل " Wolf Sheep Predation " را از قسمت Biology انتخاب کنید و دکمه " Open " را فشار دهید.

[36] Biology model
[37] Wolf sheep predation

همانطور که تب واسط کاربری[38] شامل انبوهی از دکمهها[39]، سوییچها[40]، اسلایدرها[41] و مانیتورها[42] است. این عناصر واسط به شما اجازه میدهند تا با مدل تعامل برقرار کنید. دکمهها به رنگ آبی هستند، آنها مدل را ایجاد (ستاپ)، راهاندازی و متوقف میکنند. اسلایدرها و سوییچها به رنگ سبز هستند و امکان تغییر در تنظیمات مدل را فراهم میسازند. مانیتورها و نمودارها به رنگ بژ هستند که داده را نشان میدهند.

اگر دوست دارید تصویر را بزرگتر کنید تا بهتر قابل دیدن باشد شما می توانید از منوی zoom استفاده کنید.

زمانی که شما برای بار نخست مدل را باز میکنید، شما متوجه خواهید شد که نمای مدل (نمایش گرافیکی از عاملهای مدل) خالی است (یعنی کاملا سیاه است). برای آغاز کردن مدل شما باید نخست آن را ستاپ کنید.

- دکمه "setup" را فشار دهید

چه چیزی در نمای مدل ظاهر میشود؟

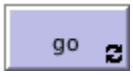
- دکمه "go" برای شروع شبیهسازی فشار دهید

همانطور که مدل در حال اجراست، چه اتفاقی برای جمعیت گوسفند و گرگ رخ میدهد؟

- دکمه "go" را برای توقف مدل فشار دهید

کنترل مدل: دکمه ها

با فشار یک دکمه، شما واکنشی را در مدل فعال میکنید. یک دکمه میتواند یک دکمه once (یکباره) یا یک دکمه forever (همیشه) باشد. نماد روی هر دکمه این تفاوت را نشان میدهد. دکمههای forever روی گوشه سمت راستشان دوتا فلش دارند. مثل این:



دکمههای once چنین فلشهای را ندارند. مثل این

[38] Interface
[39] Button
[40] Switch
[41] Slider
[42] Monitor

دکمههای once یک عمل را یکبار انجام میدهند و سپس متوقف میشوند. در چنین دکمههایی وقتی که عمل اتمام یافت سپس دکمه دوباره به حالت قبل برمیگردد.

دکمههای forever یک عمل را بارها و بارها انجام میدهند. وقتی شما بخواهید عمل را متوقف کنید، دوباره این دکمه را فشار دهید، این کار عمل فعلی را متوقف خواهد کرد و دکمه دوباره به حالت قبل برمیگردد.

بسیاری از مدلها نظیر " Wolf Sheep Predation " دارای یک دکمه once بنام " setup " و یک دکمه forever بنام "go" میباشند. مدلهای بسیار هم وجود دارد که یک دکمه once بنام " go once" یا " step once " دارند که مشابه با دکمه " go" عمل می کند با این فرق که این دکمهها بجای تکرار دایم مدل فقط یک گام زمانی[43] مدل را به جلو می برند.

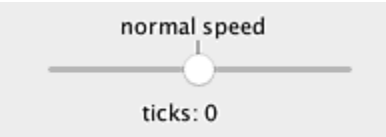
استفاده از چنین نوعی از دکمه once باعث می شود تا شما پشروی (حرکت) را از نزدیکتر مشاهده کنید (بدانید در هر گام زمانی مدل چگونه رفتار میکند).

توقف دکمه forever یک روش نرمال برای ایجاد مکث یا متوقف سازی مدل است. بعد از ایجاد مکث در مدل شما میتوانید با فشار مجدد دکمه go آن از سر بگیرید. (شما همچنین می توانید با بهرگیری از آیتم "halt " در منوی Tools مدل را متوقف کنید. اما فقط شما باید زمانی از این آیتم استفاده کنید که مدل در یک شرایط توقف ناپذیرگیرکرده باشد. استفاده از آیتم "halt" ممکن است مدل را در وسط یک عمل قطع کند و در نتیجه مدل دچار اختلال گردد)

- اگر مایل باشید میتوانید دکمههای "go" و "setup" مربوط به مدل Wolf Sheep Predation را آزمایش کنید.

آیا اگر چندین بار مدل را با تنظیمات مشابه اجرا کنید ، نتایج متفاوتی به دست می آورید؟

اسلایدر speed به شما امکان می دهد تا سرعت یک مدل را کنترل کنید ، یعنی سرعت حرکت عاملهای متحرک (تِرتِلها[44]) ،تغییر رنگ اجزای محیط (پَچها[45]) و غیره.



[43] Tick (time step)
[44] turtles
[45] patches

وقتی اسلایدر را به سمت چپ حرکت می دهید ، مدل کندتر می شود بنابراین مکث های طولانی تری بین هر تیک (گام زمانی) وجود دارد. این امر باعث می شود مشاهده آنچه اتفاق میافتد آسانتر شود. حتی ممکن است مدل را تا حدی کند کنید تا ببینید دقیقاً یک عامل متحرک(یعنی تِرتِل) چه کاری انجام می دهد.

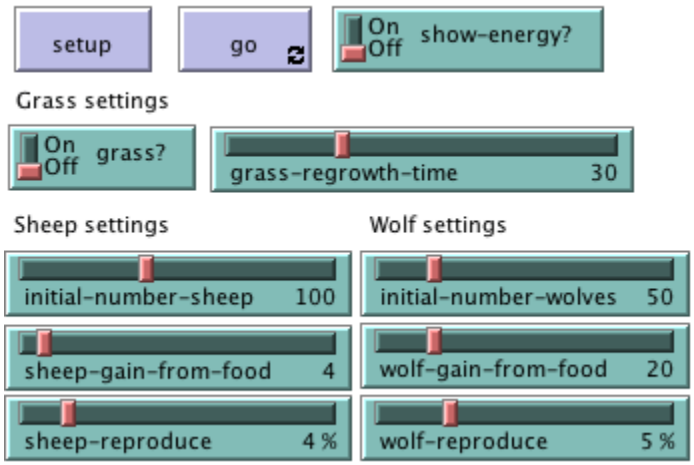
هنگامی که اسلایدر سرعت را به سمت راست حرکت می دهید سرعت مدل افزایش می یابد. NetLogo شروع به پَرش از روی فریم ها[46] میکند، یعنی در پایان هر تیک، نمای مدل را به روز نمیکند ، فقط در بعضی تیکها آن را به روز میکند. عملیات به روزرسانی زمانبر است ، بنابراین تعداد کمتری از بروزرسانیها به معنای پیشرفت سریعتر مدل است.

توجه داشته باشید که اگر اسلایدر سرعت را به سمت راست فشار دهید ، ممکن است نمای مدل به ندرت به روزرسانی شود که به نظر می رسد مدل کند شده است. در واقع این طور نیست، چون همانطورکه میبینید شمارنده تیک سریع حرکت میکندو فقط دفعات بهروزرسانی نمای مدل کمتر شده است.

کنترل سرعت: اسلایدرها و سوییچ ها

تنظیمات (setup) یک مدل به شما امکان میدهد سناریوها یا فرضیههای مختلف را بررسی کنید. تغییر تنظیمات و سپس اجرای مدل برای دیدن نحوه واکنش آن میتواند درک عمیقتری از پدیدههای مدلسازی شده به شما بدهد.

سوئیچ ها و اسلایدرها به شما امکان دسترسی به تنظیمات یک مدل را میدهند. در اینجا سوئیچ ها و اسلایدرها در Wolf Sheep Predation آورده شده است:



بیایید تاثیر آنها بر رفتار مدل را آزمایش کنیم:

- مدل Wolf Sheep Predation را باز کنید اگر از قبل باز نیست.

[46] Skipping frames

- دکمه "setup" و سپس "go" را فشار دهید و بگذارید مدل برای 100 تیک اجرا شود (شمارش تیک در بالای نمای مدل نشان داده شده است).
- با فشار دادن دکمه " go" مدل را متوقف کنید.

با گذشت زمان چه اتفاقی برای گوسفندان افتاد؟

بیایید نگاهی بیندازیم و ببینیم وقتی که ما تنظیمات را تغییر میدهیم چه اتفاقی برای گوشفندان می افتد.

- سوییچ "grass?" را روشن (on) کنید.
- دکمه "setup" و سپس "go" را فشار دهید و بگذارید مدل مطابق با حالت قبلی برای 100 تیک اجرا شود.

سوییچ چه کاری انجام داد؟ آیا نتیجه مشابه با همان اجرای قبلی شما بود؟

روشن کردن سوییچ "grass?" نتیجه مدل را تحت تاثیر قرار میدهد. با خاموش بودن سوییچ ، مقدار چمن موجود همیشه ثابت است (یعنی وقتی گوسفندان آن را خوردند مقدار آن تغییری نمیکند). این نگاه واقعبینانهای به مدل صیاد-صید نیست. چون در واقعیت چمن وقتی خورده شد با یک نرخی مجدد رشد میکند بنابراین با روشن کردن سوییچ "grass?" ما امکان رشد مجدد را به چمن میدهیم. در این حالت، ما قادر به مدلسازی هر سه عامل یعنی جمعیت گوسفند ، گرگ و چمن خواهیم بود.

اسلایدر نیز نوع دیگری از تنظیمات میباشد.

علاوه بر سوییچ، یک مدل می تواند اسلایدر هم داشته باشد. گرچه یک سوییچ فقط دو مقدار دارد (خاموش و روشن)، یک اسلایدر دامنه گسترده ای از اعداد را شامل میشود. بعنوان مثال اسلایدر " initial-number-sheep " دارای حداقلی از 0 تا حداکثری معادل 250 است. مدل می تواند با 0 گوسفند یا 250 گوسفند -یا چیزی بین این دومقدار -اجرا شود. این را امتحان کنید و ببینید چه اتفاقی می افتد. همانطور که نشانگر را از حداقل به حداکثر مقدار می برید ، شماره سمت راست اسلایدر تغییر می کند. این عددی است که در حال حاضر اسلایدر روی آن تنظیم شده است.

بیایید اسلایدرهای مدل Wolf Sheep Predation را مورد بررسی قرار دهیم.

- از تب واسط به تب اطلاعات[47] بروید و مشاهده کنید که هر اسلایدر بیانگر چه چیزی است.

تب اطلاعات راهنماییها و اطلاعات کابردی بسیار خوبی درخصوص مدل و اجزای ان ارایه میدهد. در داخل این تب، شما یک توضیح خوب از مدل، پیشنهاداتی برای آزمایش و سایر اطلاعات خواهید یافت.

[47] Info

شما شاید بخواهید قبل از اجرای مدل، تب اطلاعات را بخوانید. یا شاید شما بخواید اول مدل را آزمایش کنید و بعد تب اطلاعات آن را بخوانید.

اگر در ابتدا تعداد گوسفندان بیشتر و تعداد گرگها کمتر باشد، برای جمعیت گوسفندان چه اتفاقی می افتد؟

- سوییچ "grass?" را خاموش کنید.
- اسلایدر "initial-number-sheep?" را روی عدد 100 تنمظیم کنید.
- اسلایدر "initial-number-wolves?" را روی عدد 20 تنظیم کنید.
- دکمه "setup" و سپس "go" را فشار دهید.
- بگذارید مدل برای 100 تیک اجرا شود.

سعی کنید به دفعات زیادی مدل را با همین تنظیمات اجرا کنید.
برای جمعیت گوسفندان چع اتفاقی میافتند؟
آیا این نتیجه برای شما غافلگیر کننده بود؟ برای کمک کردن به (افزایش) جمعیت گوسفندان چه اسلایدرها و سوییچهای دیگری را میتوان تغییر داد؟

- اسلایدر"initial-number-sheep?" را روی عدد 80 و اسلایدر"initial-number-wolves?" را روی عدد 50 تنظیم کنید (این تقریباً به تنظیماتی که شما برای نخستین بار مدل را باز کردید نزدیک است)
- اسلایدر " sheep-reproduce" را روی عدد 10% تنظیم کنید.
- دکمه "setup" و سپس "go" را فشار دهید.
- بگذارید مدل برای 100 تیک اجرا شود.

این دفعه چه اتفاقی برای جمعیت گرگها افتاد؟

وقتی مدلی را باز میکنید ، همه اسلایدرها و سوییچها روی تنظیمات پیش فرض قرار دارند. اگر مدل جدیدی را باز کنید یا از برنامه خارج شوید ، تنظیمات تغییر یافته شما ذخیره نمی شوند ، مگر اینکه شما بخواهید آنها را ذخیره کنید.

(توجه: علاوه بر اسلایدر و سوییچ برخی از مدلها دارای عناصر واسطی نظیر انتخابگر و جعبه ورودی می باشند که مدل Wolf Sheep Predation هیچکدام از اینها را ندارد.)

گردآوری اطلاعات: نمودارها و مانیتورها

نمای مدل به شما این امکان را میدهد که آنچه در مدل اتفاق میافتد را ببینید. افزون بر این، Netlogo راهای دیگری برای ارایه اطلاعات در خصوص مدل را نیز دارد که نمودارها و مانتیورها از جملهی آنها هستند.

نمودارها

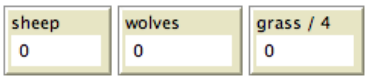
نمودار مربوط به مدل Wolf Sheep Predation شامل سه خط از جمعیت گوسفندها، گرگها و حجم علف بر 4 میباشد (مقدار علف به خاطر این بر 4 تقسیم شده است که نمودار خیلی دراز نشود). خطوط نشان دهندهی تغییراتی هستند که در طول زمان در مدل اتفاق میافتد. شرح[48] نمودار نشان میدهد که هر خط چه چیزی را نشان میدهد. در این مورد، نشان دهنده تعداد جمعیت میباشد.

هنگامی که یک نمودار نزدیک به پر شدن می شود ، محور افقی جمع میشود و تمام داده های قبلی به فضای کوچکتری فشرده می شوند. در این حالت، فضای بیشتری برای رشد نمودار فراهم میشود.

اگر میخواهیید داده یک نمودار را استخراج کنید و در فرمت دیگری ذخیره و تحلیل نمایید. از آیتم " Export Plot" در منوی File استفاده کنید. این دیتای نمودار را در فرمتی نظیر Excel ذخیره سازی میکنند که بعدها میتوان آن را خواند و تحلیل نمود. شما همچنین می تواند با راست کردن کلیک روی نمودار و انتخاب گزینه "Export..." از منوی بالاآمده این کار را نیز انجام دهید.

مانیتورها

مانتیورها ابزار دیگری برای نمایش اطلاعات از مدل هستند. مانتیورهای Wolf Sheep Predation نشان داده شدهاند.



مانتیورها جمعیت گوسفند، گرگ و حجم علف را نشان میدهند (توجه داشته باشید که حجم علف بر عدد 4 تقسیم شده است).

اعداد نمایش داده شده در مانیتورها با اجرای مدل تغییر میکنند، در حالیکه نمودارها دادهی کل دوره اجرای مدل را به شما نشان می دهند.

کنترل کردن نمای مدل

در تب واسط، شما یک نوار ابزاری از کنترلها را میبینید. برخی از آنها جنبههای از نمای مدل را کنترل میکنند.

بیایید اثر این کنترلها را آزمایش کنیم

- دکمه "setup" و سپس "go" را فشا دهید تا مدل اجرا شود.
- همانطور که مدل اجرا میشود، اسلایدر speed (بالای نمای مدل) را به سمت چپ حرکت دهید.

چه اتفاقی میافتد؟

[48] legend

اگر سرعت مدل آنقدر زیاد است که شما نمیتوانید ببینید چه چیزی دقیقاً اتفاق میافتند، این اسلایدر برای شما بسیار مفید خواهد بود.

- اسلایدر را به وسط حرکت دهید.
- حرکت اسلایدر را به سمت راست ادامه دهید.
- اکنون تیک " view updates" را فعال و غیرفعال کنید.

چه اتفاقی میافتد؟

اگر شما مشتاقید که بدون فوت وقت اجرای مدل را به صورتی خیلی سریعتر مشاهده کنید شما میتوانید با برداشتن تیک از view updates آن را خاموش کنید که از این کار بعنوان پیشروی سریع[49] یاد میشود. عمل پیشروی سریع (حرکت دادن اسلایدر speed به قسمت راست) باعث لحاظ ننمودن بهروزرسانیهای نمای مدل میشود و چون بروزرسانیهای نمای مدل کار زمانبری است این زمان صرف خود اجرای مدل میشود و با اینکار سرعت اجرا به نحوی چشمگیر افزایش مییابد.

وقتی که view updates کاملاً خاموش است، مدل در پس زمینه اجرا میشود و فقط نمودارها و مانیتورها کماکان بهروزرسانی میشوند. ولی اگر شما میخواهید مشخصاً متوجه بشوید چه چیزی اتفاق میافتد باید تیک view update را مجدد فعال کنید. هنگامی که view update خاموش است (تیک آن فعال نیست) بسیاری از مدلها سریعتر اجرا میشوند. اما در برخی مدلها این کار تفاوت اندکی ایجاد میکند.

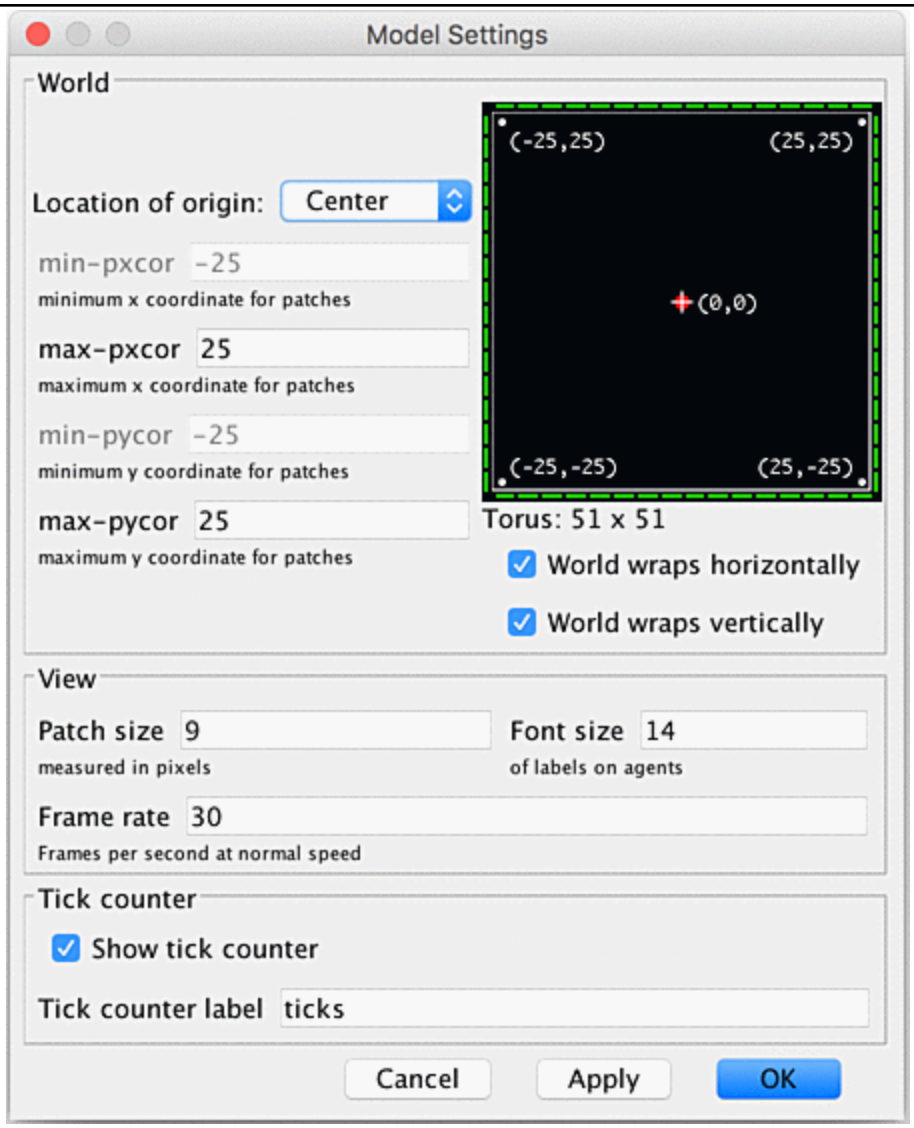
اندازه نمای مدل توسط پنج تنظیم جداگانه تعیین میشود که شامل min-pxcor، max-pxcor، min-pycor، max-pycor و patch-size تعیین میشود. بیایید ببینم چه اتفاقی میافتند وقتی شما اندازه نمای مدل را تغییر میدهید.

آنگونه که شما میبیند یکی از اجزای نوار ابزار دکمه "settings" است. این دکمه به ما امکان دسترسی به تنظیمات نمای مدل را میدهد

- دکمه " Settings" را در نوار ابزار فشار دهید.

یک دیالوگی باز خواهد شد که شامل تمام تنظیمات برای نمای مدل است.

[49] fast forwarding



تنظیمات فعلی برای max-pycor، min-pycor، max-pxcor، min-pxcor و patch-size چه هستند؟

- بدون اینکه تغییری در نمای مدل ایجاد کنید دکمه " cancel" را فشار داده و خارج شوید.
- نشانگر ماوس را در مجاورت-ولی هنوز خارج از- نمای مدل قرار دهید

متوجه خواهید شد که نشانگر crosshair به شکل علامت + در میآید.

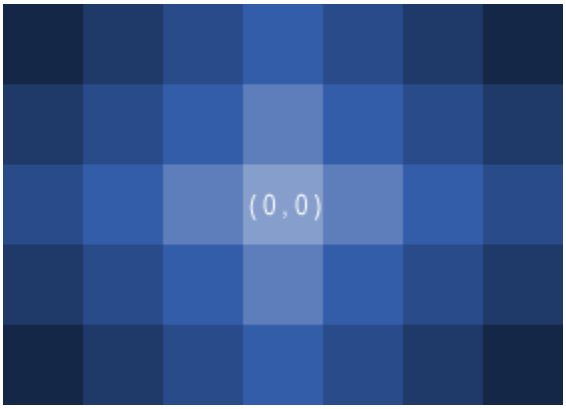
- دکمه ماوس (کلیک چپ) را نگه دارید و crosshair را به روی نمای مدل بکشید.

نمای مدل اکنون انتخاب شده است چون همانطور که میبینید با مرز خاکستری احاطه شده است که در آن 8 مربع سیاه (بنام هَندل[50]) به چشم میخورد

[50] handle

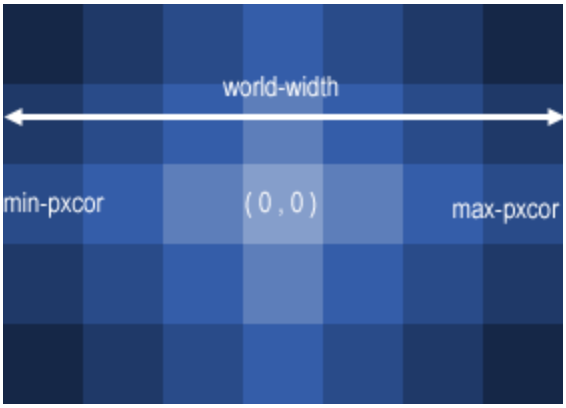
ماوس را روی یک از این هندلها بگذارید، سپس دکمه (کلیک چپ) ماوس را نگه داشته و هندل را به سمت بیرون بکشید(تاجایی که دلتان میخواهد).
با کلیک کردن بر روی هر قسمتی از پس زمینه سفید تب واسط، انتخاب نمای مدل را لغو کنید.
دکمه "Settings" در نوارابزار را دوباره فشار دهید و به تنظیمات آن نگاه کنید.
چه اعدادی تغییر کردهاند؟
چه اعدادی تغییر نکردهاند؟

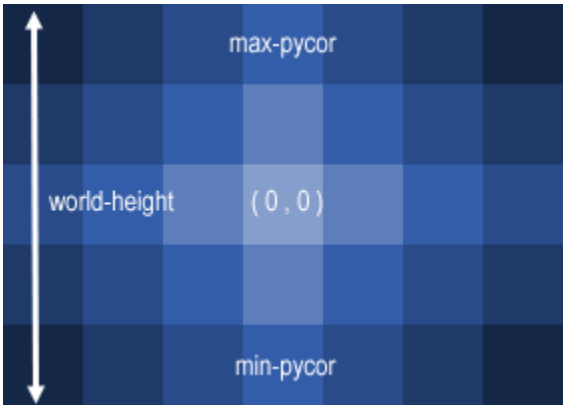
NetLogo یک محیط دوبعدی مشبک از عناصری بنام پَچ (patch) است. هر پَچ یک مربع واحد از یک فضای مشبک است. در مدل Wolf Sheep Predation، وقتی که سوییچ "grass?" روشن است. پَچهای واحد را به آسانی میشود مشاهده کرد. زیرا برخی سبز و برخی قهوهای اند. یک اتاقی که کف آن کاشی است را در نظر کنید و تصور کنید که هر پَچ حکم یک کاشی مربعی شکل از این اتاق را دارد. دقیقاً در وسط اتاق، کاشی (0,0) قرار دارد. با این تفسیر میتوان گفت که ما اکنون دستگاه مختصاتی داریم که میتواند موقعیت هر شی در اتاق را نشان دهد.



از سمت راست اتاق تا کاشی (0,0) چه تعداد کاشی وجود دارد؟
از سمت چپ اتاق تا کاشی (0,0) چه تعداد کاشی وجود دارد؟

در NetLogo ، از تعداد کاشیهای از چپ به راست (یا برعکس) بعنوان world-width نامیده میشود و از تعداد کاشیهای از بالا به پایین (یا برعکس) بعنوان world-height نامیده میشود. این اعداد توسط کرانها بالا، پایین، چپ و راست تعریف میشوند.





در این نمودارها، max-pxcor برابر با 3، min-pxcor برابر با 3-، max-pycor برابر با 2 و min-pycor برابر با 2- میباشد.

وقتی که شما patch size (اندازه پَچ) را تغییر میدهید، تعداد پَچها (کاشیها) تغییر نمیکند و فقط اندازه انها در نمای مدل بزرگتر یا کوچکتر میشود.

بیایید به تأثیر تغییر حداقل و حداکثر مختصات در جهان بپردازیم.

- از دیالوگ "Settings" که کماکان باز است استفاده کنید (اگر آن را بستهاید مجدد باز کنید)، حال max-pxcor را به 30 و max-pycor را به 10 تغییر دهید و سپس Ok را فشار دهید. توجه داشته باشید که min-pxcor و min-pycor به صورت خودکار تغییر میکنند. دلیل این تغییر هم این است که به صورت پیش فرض مبدأ (0,0) بعنوان مرکز محیط (جهان) در نظر گرفته شده است.

برای شکل نمای چه اتفاقی مدل افتاد؟

- دکمه "setup" را فشار دهید.

حال شما میتوانید پَچهای جدیدی که ایجاد کردهاید را ببینید

- با فشار دادن مجدد دکمه "Settings..." نمای مدل را ویرایش کنید.

- اندازه پَچ (patch-size) را به 20 تغییر دهید و دکمه "OK" را بزنید.

ویرایش نمای مدل به شما امکان میدهد تا سایر تنظیمات را نیز تغییر دهید. بدون هیچ محدودیتی سعی کنید سایر تنظیمات را نیز خودتان آزمایش کنید.

هنگامی که شما مدل Wolf Sheep Predation را بررسی کردید شاید شما دوست داشته باشید سایر مدلهای موجود در کتابخانه مدلها را نیز بررسی کنید.

کتابخانه مدلها

این کتابخانه شامل چهار بخش: مدلهای نمونه[51]، مدلهای برنامه درسی[52]، مثالهای کد[53] و فعالیتهای HubNet میباشد.

مدلهای نمونه

بخش مدلهای نمونه بر اساس موضوع تنظیم شده است و در حال حاضر بیش از 200 مدل را شامل می شود. ما به طور مداوم در حال کار بر روی اضافه کردن مدلهای جدید به این بخش هستیم ، بنابراین برای مشاهده موارد اضافی جدید در کتابخانه همیشه سعی کنید هر چند وقت یکبار به این بخش سر بزنید.

برخی از پوشه های موجود در مدلهای نمونه دارای پوشه هایی با عنوان "(تأیید نشده[54])" هستند. این مدل ها کامل و کاربردی میباشند، اما هنوز در مرحله بررسی محتوا ، صحت و کیفیت کد هستند.

مدل های برنامه درسی

این مدلها را مرکز یادگیری متصل (CCL) دانشگاه نورثوسترن طی یک برنامه درسی، برای کاربرد در مدارس توسعه داده است. برخی از این مدل در بخش مدلهای نمونه نیز وجود دارند ولی برخی دیگر کاملا منحصر این بخش میباشند. سعی کنید تب اطلاعات این مدلها را مطالعه کنید تا اطلاعات بیشتری در خصوص کابرد انها پیدا کنید.

مثالهای کد

این بخش شامل مثالهای سادهای از مشخصههای خاص NetLogo است. بعدها که شما خواستید مدلهای موجود را توسعه دهید یا حتی مدل جدید خودتان را بسازید این مثالها به شدت بدردتان خواهند خورد. بعنوان نمونه اگر شما بخواهید یک هیستوگرام به مدلتان اضافه کنید شما میتوانید نگاهی به "Histogram Example" داشته باشید و یاد بگیرید که چگونه اینکار را انجام دهید

[51] Sample Models
[52] Circular Models
[53] Code Examples
[54] Unverified

فعالیت های HubNet

این بخش شامل شبیه سازیهای مشارکتی برای استفاده به صورت گروهی است . برای کسب اطلاعات بیشتر در مورد HubNet، به راهنمای HubNet مراجعه کنید.

آنچه در ادامه خواهد آمد

اگر میخواهید یادبگیرید که چگونه مدلها را در سطح عمیقتری مورد کاوش قرار دهید، آموزش شماره2: فرمانها شما را با زبان برنامهنویسی NetLogo آشنا خواهد ساخت.

در آموزش شماره 3: رویهها شما خواهید آموخت که چگونه مدلهای موجود را تغییر و توسعه داده و مدل جدید خودتان را بسازید.

آموزش شماره 2: فرمانها

شما تاکنون با موفقیت مراحل بازکردن و اجرای مدلها ، فشار دادن دکمهها، تغییر اسلایدرها و سوییچ ها و جمع آوری اطلاعات از یک مدل با استفاده از نمودارها و مانیتورها، را انجام دادهاید.

در این بخش، تمرکز از مشاهده مدلها به دستکاری مدلها تغییر میکند. در این بخش شما شروع به فهمیدن نحوه کار مدل و چگونگی تغییر آن خواهیدکرد.

- مدل نمونه: مدل پایه ترافیک
- مرکز فرمان
- کارکردن با رنگها
- مانیتورهای عامل و فرماندههای عامل

مدل نمونه : مدل پایه ترافیک

- از منوی File گزینه Models Library را بازکنید.
- از بخش "Social Science" مدل "Traffic Basic" را بازکنید.
- مدل را برای لحظاتی اجرا کنید تا یک دید کلی نسبت به آن پیدا کنید
- برای هر سوالی که برایتان مطرح است از تب اطلاعات کمک بگیرید.

در این مدل، شما یک اتومبیل قرمز را در جریانی از اتومبیلهای آبی مشاهده میکنید. این جریان اتومبیلها به طور کلی در یک جهت حرکت میکند. همانطورکه میبینید هراز چندگاهی آنها "جمع میشوند" و از حرکت میایستند. این نشان میدهد که چگونه چیزی مانند ترافیک (راهبندان) میتواند بدون دلایل خاصی نظیر تصادف شکل گیرد.

شما میتوانید تنظیمات را به طور دلخواه تغییر داده و طی چند اجرای مختلف درک کاملتری از مدل بدست آورید.

همانظور که دارید از مدل Traffic Basic استفاده میکنید، آیا به مواردی اندیشیدهاید که مایلید به مدل اضافه کنید ؟

با نگاه به مدل Traffic Basic ، شاید متوجه شده باشید که محیط آن تقریباً ساده است با پس زمینهای سیاه، خیابانی سفید و تعدادی ماشین آبی و یک ماشین قرمز. (برخی) تغییراتی که میتوان در مدل اضافه کرد عبارتند از: تغییر رنگ و شکل اتومبیلها، افزودن یک خانه یا چراغ خیابان، ایجاد یک چراغ توقف یا حتی ایجاد لاین دیگری از ترافیک. برخی از این تغییرات پیشنهادی بصری هستند که نما و ظاهر مدل را تغییرمیدهند در حالی که برخی دیگر بیشتر جنبه رفتاری دارند. در قسمت آموزشی این مدل ما بیشتر بر تغییرات سادهتر و ظاهری تمرکز میکنیم (آموزش شماره 3 با جزییات بیشتری به تغییرات رفتاری میپردازد زیرا آنها مستلزم تغییرات در کد مدل میباشند)

برای ایجاد این تغییرات ساده ما از مرکز فرمان استفاده میکنیم.

مرکز فرمان

مرکز فرمان[55] را میتوان در تب واسط پیدا کرد. این به شما امکان میدهد تا فرمانها یا جهتهایی به مدل بدهید. فرمانها دستورالعملهای هستند که شما میتوانید برای عاملهای NetLogo یعنی تِرتِلها، پَچها و لینکها و حتی مشاهدهگر بنوسید.

در مدل Traffic Basic

- دکمه "Setup" را فشار دهید.
- به مرکز فرمان بروید
- در کادر سفید پایین مرکز فرمان کلیک کنید[56].
- متنی که در اینجا نشان داده شده را تایپ کنید.

- دکمه بازگشت (enter) را فشار دهید

در نمای مدل چه اتفاقی میافتد؟

شاید متوجه شدید که تمام پس زمینه محیط به رنگ زرد درآمده است و خیابانها ناپدید شدند.

پس چرا رنگها اتومبیلها زرد نشد؟

دوباره به فرمانی که نوشته بودید نگاه کنید، ما از پَچها خواستیم که زرد شوند (نه چیز دیگر). در این مدل، اتومبیلها توسط عاملهای متفاوتی بنام "تِرتِل" نشان داده شدهاند. بنابراین، اتومبیلها این دستورالعملها را دریافت نمیکنند و از اینرو رنگشان نیز تغییر نمیکند.

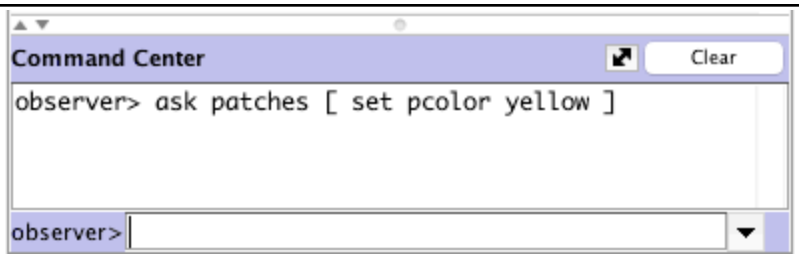
چه تغییری در مرکز فرمان رخ داد؟

شاید متوجه شده باشید فرمانی که شما تایپ کردید الان به صورت زیر در مرکز فرمان نشان داده شده است.

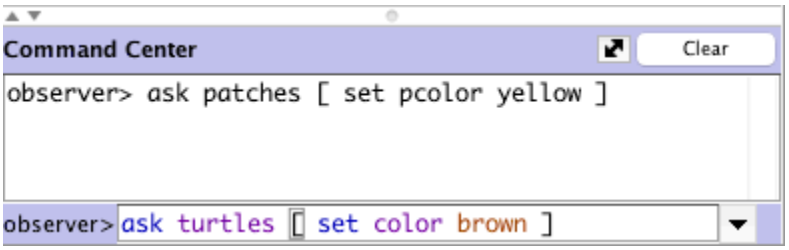
---

[55] Command Center

[56] برای اطلاع یافتن از کارکرد هر سینتَکسی از NetLogo شما فقط باید آن را در مرکز فزمان (observer> syntax name) تایپ کنید و سپس دکمه F1 را فشار دهید. این کار شما را به دیکشنری NetLogo برده و اطلاعات کاربردی مفیدی را در خصوص سینتَکس مورد نظر ارایه میدهد (مترجم).



- در انتهای مرکز فرمان، متن زیر را تایپ کنید.



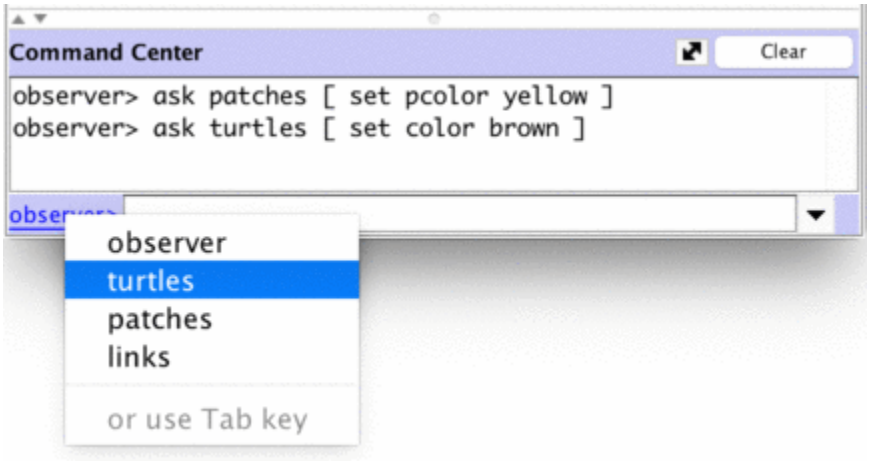
آیا نتیجه مورد انتظار شما بود؟

نمای مدل اکنون دارای پس زمینهای زرد رنگ و تعدادی ماشین به رنگ قهوه‌ای است.

جهان NetLogo یک جهان دوبعدی است که از تِرتِلها، پَچها، لینکها و مشاهدهگر تشکیل شده است. پَچها بستری هستند که تِرتِلها روی آنها حرکت میکنند. لینکها اتصالات بین تِرتِلها هستند و مشاهدهگر همان عامل خارجی است که بر آنچه در مدل رخ میدهد نظارت دارد (برای جزییات بیشتر به بخش **راهنمای برنامهنویسی NetLogo** مراجعه کنید)

در مرکز فرمان شما میتوانید به هر یک از این عاملها دستور دهید. با انتخاب منوی موجود در گوشه سمت چپ کادر سفید رنگ، شما میتوانید نوع عامل (تِرتِلها، پَچها، لینکها یا مشاهدهگر) را انتخاب کنید. افزون بر این شما از کلید tab موجود در صفحه کلیدتان هم میتوانید استفاده کنید و یکی از این عاملها را انتخاب کنید.

- در گوشه سمت چپ مرکز فرمان، روی "observer>" کلیک کنید.



- نوع "turtles" را از منوی حاصله انتخاب کنید.
- عبارت set color pink را تایپ کنید و دکمه بازگشت (enter) را فشار دهید.
- روی کلید tab چندباری فشار دهید تا زمانی که "patches> " را در گوشه سمت چپ مشاهد کنید.
- عبارت set pcolor white را تایپ کنید و دکمه بازگشت (enter) را فشار دهید.

اکنون نمای مدل چه شکلی شده است؟

آیا بین این دو فرمان و فرمانهایی که قبلا از سمت observer اعمال می شود تفاوتی وجود دارد؟

مشاهدهگر بر جهان نظارت دارد و با استفاده از دستور ask می تواند به پَچها و تِرتِلها فرمان دهد. همانند مثال اول (Observer > ask patches [set pcolor yellow]) مشاهدهگر باید از پَچها بخواهد تا رنگشان را به سبز تغییر دهند. اما وقتی یک دستور مستقیم به گروهی از عاملها داده میشود مانند مثال دوم (patches > ask set pcolor yellow) شما فقط باید خود دستور را بدهید.

- دکمه "setup" را فشار دهید؟

چه اتفاقی افتاد؟

چرا نمای مدل به حالت قبل برگشت یعنی زمانی که پس زمینه مشکی و راه سفید بود؟ با فشار دکمه "setup"، مدل خودش را بر اساس تنظیمات تشریح شده در تب کد[57]، پیکربندی مجدد مینماید. مرکز فرمان تغییرات دایمی در مدل ایجاد نمیکند. مرکز فرمان در واقع به شما امکان می دهد تا با دستکاری کردن مدل، فرضیاتی که در حین اجرای مدل به ذهنتان میرسد را بیازمایید(تب کد در آموزش بعدی و بخش راهنمای برنامهنویسی تشریح شده است).

حال که با مرکز فرمان آشنا شدهاید، خوب است با نحوه کار رنگها در نتلوگو بیشتر آشنا شویم.

کارکردن با رنگها

در بخش پیش شاید شما متوجه شدهاید که ما از دو کلمه متفاوت color و pcolor برای تغییر رنگها استفاده کردیم.

بین color و pcolor چه تفاوتی وجود دارد؟

- از منوی حاصله در مرکز فرمان "turtles" را انتخاب کنید (با کلید tab هم می توانید)
- عبارت set color blue را تایپ کرده و دکمه برگشت (enter) را فشار دهید.

چه اتفاقی برای اتومبیلها میافتد؟

به این فکر کنید که برای آبی شدن اتومبیل ها چه کاری انجام داده اید و سعی کنید که پَچها را هم قرمز شود.

اگر شما از پَچها بخواهید که عبارت set color red، یک پیام خطا صادر خواهد شد.

- در عوض عبارت set pcolor red را تایپ کنید و دکمه برگشت را بزنید.

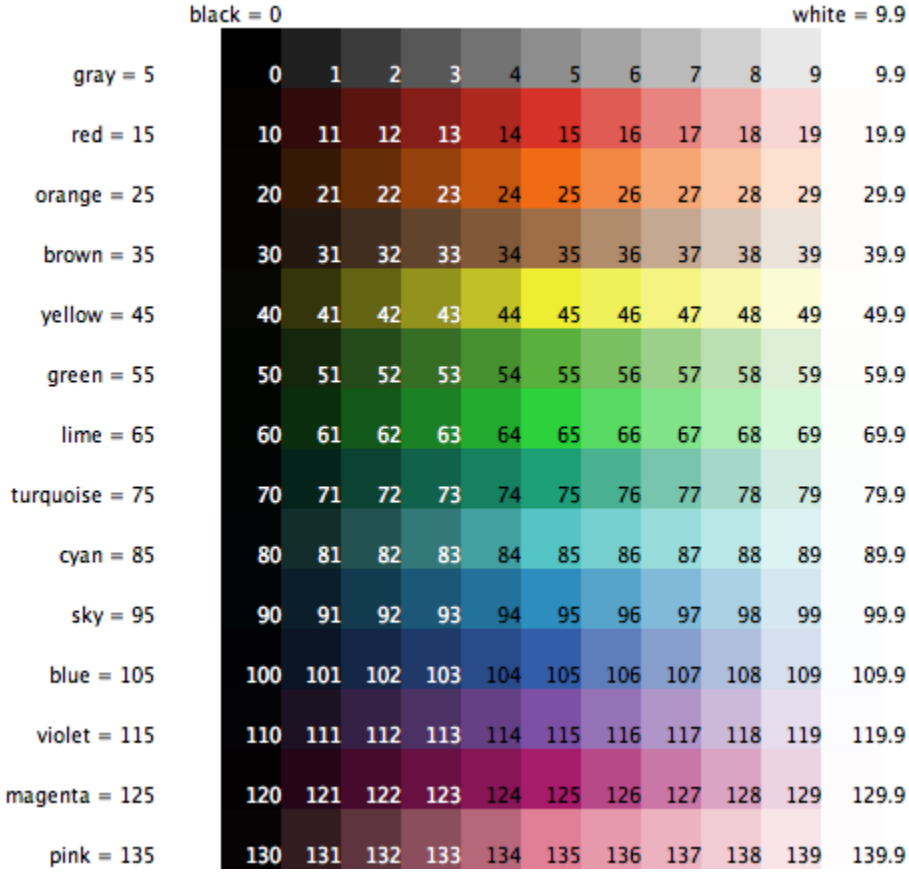
ما از color و pcolor بعنوان متغیر یاد میکنیم. برخی از فرمانها و متغیرها فقط خاص تِرتِلها هستند و برخی هم فقط مختص پَچها میباشند. بعنوان نمونه، متغیر color یک متغیر مربوط به تِرتِل می باشد در حالیکه متغیر pcolor یک متغیر مربوط به پَچها میباشد.

ادامه دهید و تغییر رنگ تِرتِلها و پَچها را با استفاده از دستور set و این دو متغیر تمرین کنید.

برای اینکه بتوانیم تغییرات بیشتری را در رنگ تِرتِلها و پَچها، یا بعبارتی ماشینها و محیط ایجاد کنیم لازم است تا نسبت به رنگهای NetLogo اطلاعات بیشتری بدست آوریم. در NetLogo رنگها دارای مقادیر عددی هستند. در تمام مثالهایی که تاکنون انجام دادیم ما از نام رنگها استفاده میکردیم. دلیل این موضوع آن است که NetLogo فقط از 16 نام رنگ استفاده میکند. این بدان معنا نیست که NetLogo فقط 16 رنگ را

[57] Code

تشخیص میدهد. سایه های[58] زیادی در بین این رنگ ها وجود دارد که میتواند مورد استفاده قرار گیرد. در ادامه یک نموداری ارایه شده است که کل فضای رنگ NetLogo را نشان می دهد.



فرض کنید شما بخواهید یک رنگی بگیرید که چیزی شبیه قرمز کم رنگ میباشد (یعنی یک سایه میانی) شما می توانید به اضافه کردن یا کاستن یک عدد از رنگ قرمز به این رنگ دست یابید. بعنوان نمونه شما برای ایجاد رنگ قرمز برای یک تِرتِل میتوانید عبارت set color red یا حتی set color 15 را تایپ کنید. با استفاده از یک عدد که کمی بزرگتر یا کمی کوچکتر است ، شما می توانید یک نسخه روشنتر یا تیره تری از همان رنگ دریافت کنید . این عملیات به شرح زیر میباشد:

- از منوی حاصله در مرکز فرمان "patches" را انتخاب کنید (با کلید tab هم می توانید)
- عبارت set pcolor red - 2 را تایپ کنید (توجه کنید فاصله اطراف علامت "-" مهم است)

با کاستن از رنگ قرمز شما آن را تیرهتر میکنید

- عبارت set pcolor red + 2 را تایپ کنید

[58] shade

با افزودن به رنگ قرمز شما ان را روشنتر میکنید

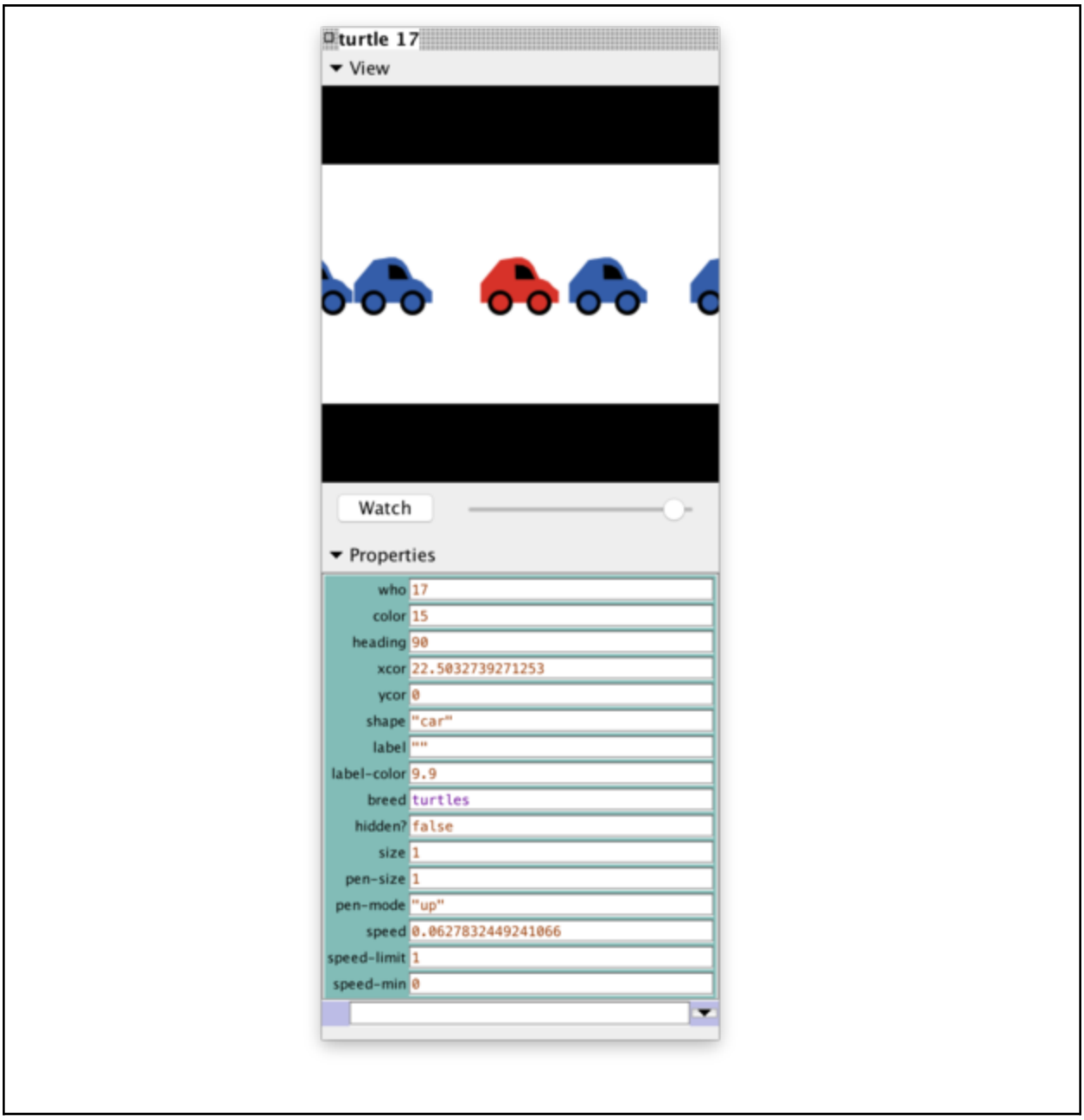
.شما میتوانید این تکنیک را روی هر رنگی که در نمودار فوق آمده است پیاده سازی کنید

مانیتورهای عامل و فرماندههای عامل

در فعالیت قبلی، ما از فرمان set برای تغییر رنگ اتومبیلها استفاده کردیم. اما اگر بیادتان باشد، در مدل اصلی در بین تعدادی از ماشینهای آبی، فقط یک ماشین قرمز رنگ وجود داشت. بیایید ببینیم که چگونه میشود فقط رنگ یک ماشین را تغییر داد.

- دکمه "setup" را فشار دهید تا ماشین قرمز مجدد ظاهر شود.
- روی ماشین قرمز کلیلک راست کنید.
- یک منویی ظاهر می شود که در انتهای آن یک تِرتِل بهراه شمارهاش وجود دارد. توجه کنید اگر در کنار ماشین قرمز چند ماشین باشند ممکن است چند تِرتِل را مشاهده کنید. شما روی هر تِرتِلی که ماوس را بگذارید ان تِرتِل برجسته (هایلایت) میشود و میفهمید کدام تِرتِل ماشین قرمز زنگ میباشد و بعد از انتخاب تِرتِل مورد نظر، یک زیر منو ظاهر می شود که از میباید گزینه " inspect turtle " را انتخاب کنید.

یک مانیتور تِرتِل به صورت زیر پیدا خواهد شد:



نمای کوچکی که در بالای مانیتور عامل (در اینجا تِرتِل) وجود دارد همراه با مرکزیت این عامل باقی خواهد ماند. با استفاده از اسلایدر زیر نما شما میتوانید نما را بزرگ و کوچک کنید و با فشار دادن دکمه "watch-me" می توانید این تِرتِل را در نمای اصلی مدل تماشا کنید.

بیایید نگاهی دقیقتر به مانتیور تِرتِل داشته باشیم.

شماره هویتی[59] تِرتِل چیست؟
رنگ این تِرتِل چیست؟
این تِرتِل چه شکلی دارد؟

این تِرتِل دارای شماره هویتی 0 است، رنگ آن 15 (رنگ قرمز مطابق چارت فوق) و شکل آن اتومبیل است.

افزون بر راست کلیک کردن روی یک تِرتِل، دو راه دیگر برای باز کردن مانتیور تِرتِل وجود داارد. یک راه این است که از منوی ابزار یعنی "Tools menu" گزینه "Turtle Monitor" را انتخاب کنید سپس شماره هویتی آن را در فیلد مربوطه تایپ کنید و دکمه enter را فشار دهید. راه دیگر این است که در مرکز فرمان عبارت inspect turtle 0 (یا هر شماره دیگری) را تایپ کنید.

برای بستن مانیتور تِرتِل، شما می تواند روی علامت ضربدر در گوشه بالای سمت چپ (اگر از سیستم Mac) یا در گوشه بالایی سمت راست (سایر سیستم های عامل) کلیک کنید

حال که در خصوص مانیتور عامل اطلاعات بیشتری بدست آوردیم ما سه راه برای تغییر رنگ هر تِرتِل داریم.

یک راه این است که از کادر فرمانده عامل[60] استفاده کنیم که در پایین مانیتور عامل وجود دارد در اینجا شما میتوانید فرمان خودتان را تایپ کنید و نتیجه را ببینید دقیقا مشابه با کاری که در مرکز فرمان انجام میدادیم با این فرق که فرمان هایی که اینجا می دهدی فقط توسط همان تِرتِل بخصوص اجرا میشوند.

- در تِرتِل مانیتور مربوط به تِرتِل صفر، به قسمت انتهایی یعنی کادر مرکز فرمان آن تِرتِل بروید و عبارت set color pink را تایپ و اجرا کنید.

چه اتفاقی در نمای مدل افتاد؟
آیا تغییری در تِرتِل مانیتور رخ داد؟

راه دوم برای تغییر رنگ یک تِرتِل این است که مستقیماً متغیر رنگ در تِرتِل مانیتور را دستکاری کنید (نام عدد رنگ مورد نظر را در آن قرار دهید)

- در تِرتِل مانیتور عدد سمت راست رنگ را انتخاب کنید
- عبارت green + 2 را تایپ و سپس اجرا کنید.

چه اتفاقی افتاد؟

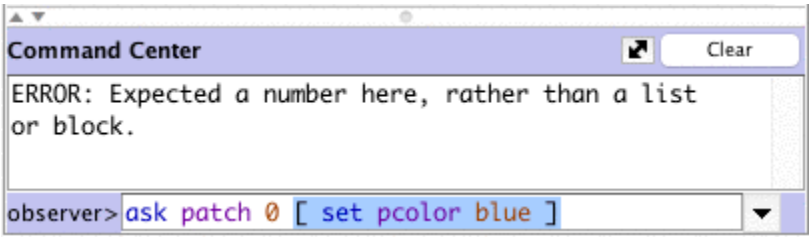
[59] who
[60] Agent Commnder

راه سوم برای تغییر رنگ یک تِرتِل یا پَچ این است که به مرکز فرمان (پایین نمای مدل) بروید و بعنوان مشاهدهگر این عمل را انجام دهید. چون مشاهدهگر ناظر بر جهان است میتواند فرمانهایی بدهد که نتنها هر تِرتِل را به صورتی فرد بلکه گروهای از انها را تحت تاثیر قرار دهد.

- در مرکز فرمان، گزرینه"observer" را از منوی حاصله انتخاب کنید (از کلید tab هم میتوانید استفاده کنید)
- عبارت ask turtle 0 [set color blue] را تایپ و سپس اجرا کنید.

چه اتفاقی افتاد؟
همانطور که برای تِرتِلها مانیتور وجود دارد برای پَچها (حتی لینکها) نیز مانیتور وجود دارد. که عملکرد آنها بسیار شبیه مانیتور تِرتِلها است.
آیا میتوانید یک مانیتور پَچ ایجاد کنید و از طریق آن رنگ پَچ مربوطه را تغییر دهید؟

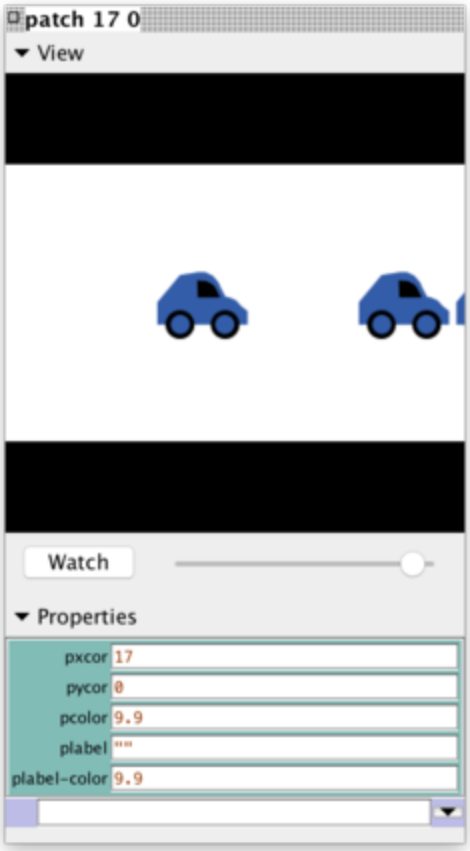
اگر شما در مرکز فرمان و بعنوان مشاهدهگر عبارت ask patch 0 [set pcolor blue] را اجرا کنید شما با خطای زیر مواجه خواهید شد.



برای اینکه به صورت مشخص از یک تِرتِل بخواهیم تا یک کاری انجام دهد، ما از شماره هویتی آن استفاده میکردیم. ولی پَچها شماره هویتی ندارند از اینرو به طریقی دیگر باید به آنها رجوع کنیم.

بیاد بیاورید که پَچها در یک دستگاه مختصات مرتبسازی شدهاند. برای مشخص کردن یک نقطه در یک نمودار به دو عدد (مؤلفه) بنام های مقدار در محور x و مقدار در محور y نیاز است. مکان هر پَچ هم مشابه با این روش مشخص میشود.

- برای یک پَچ، مانیتور آن را باز کنید (روی آن راست کلیک کنید و انتخاب " inspect patch " کنید)



مانیتور نشان می دهد که مولفه اول پَچ یعنی pxcor آن برابر با 17 و مولفه دوم پَچ یعنی pycor آن برابر با 0 است. که هر دوی این مؤلفه ها همان x و y محور مختصات میباشند.
برای اینکه به این پَچ بگویید رنگش را تغییر بدهد:

- در انتهای مانیتور پَچ، عبارت set pcolor blue را تایپ و اجرا کنید.

تایپ کردن فرمان در مانیتور هر تِرتِل یا پَچ فقط آن تِرتِل یا پَچ را مخاطب قرار میدهد

- شما همچنین میتوانید از مرکز فرمان نیز به یک تِرتِل یا پَچ مشخص دستور دهید
- در مرکز فرمان، عبارت ask patch 17 0 [set pcolor green] را تایپ و اجرا کنید .

آنچه در ادامه خواهدآمد

در این مرحله شاید شما بخواهید از تکنیکهایی که آموختهاید در سایر مدلهای موجود در کتابخانه مدلها استفاده کنید.

در بخش **آموزش شماره 3: رویهها** شما خواهید آموخت که چگونه مدلهای موجود را تغییر و گسترش داده و مدل خودتان را ایجاد کنید.

آموزش شماره3: رویه ها

طی این آموزش، شما مرحله به مرحله با فرایند ساخت یک مدل کامل آشنا خواهید شد.

- عاملها و رویهها[61]
- ایجاد دکمه setup
- تغییر به روزرسانیهای مبتنی بر تیک[62]
- ایجاد دکمه go
- آزمایش با فرمانها
- پَچها و متغیرها
- متغیرهای تِرتِل
- مانیتورها
- سوییچها و برچسبها[63]
- رویههای بیشتر
- ترسیم نمودار
- شمارنده تیک[64]
- برخی جزییات بیشتر
- آنچه در ادامه خواهد آمد
- ضمیمه: کد کامل

عاملها و رویه ها

در آموزش شماره 2، شما یادگرفتید که چگونه از مرکز فرمان و مانیتورهای عاملها برای بررسی و تغییر عامل ها و همچنین دستور دادن به عاملها استفاده کنید. اکنون شما آماده اید تا قلب واقعی NetLogo، یعنی تب کد[65] را بیاموزید.

ما میدانیم که در NetLogo چهار نوع عامل مختلف بنامهای پَچها، تِرتِلها، لینکها و مشاهدهگر وجود دارد. پَچها آن اجزای ثابتی میباشند که در یک فضای مشبک چیده شدهاند. تِرتِلها روی پَچها حرکت میکنند. لینکها

[61] Procedure
[62] Switching to tick-based view updates
[63] Label
[64] Tick counter
[65] Code tab

دو تِرتِل را بهم وصل میکنند. مشاهدهگر بر اتفاقی که در NetLogo رخ میدهد نظارت میکند و هر آنچه پَچها، تِرتِلها، لینکها نمیتوانند برای خودشان انجام دهد را انجام میدهد.

هر چهارنوع از عاملها میتوانند فرمانهای NetLogo را اجرا کنند. هر چهار نوع قادرند "رویهها" را نیز اجرا کنند. یک رویه مجموعهای از فرمانهای NetLogo را در یک فرمان جدید واحد که شما تعریف میکنید، ترکیب میکند.

اکنون خواهید آموخت که رویههایی را بنویسید که باعث میشوند تِرتِلها حرکت کنند، بخورند، تولید مثل کنند و حتی بمیرند. شما همچنین خواهید آموخت که چگونه مانیتور، اسلایدر و نمودار بسازید. مدلی که ما در این بخش توسعه میدهیم یک اکوسیستم ساده است و به اندازه مدل Wolf Sheep predation بخش آموزش شماره 1 پیچیده نیست.

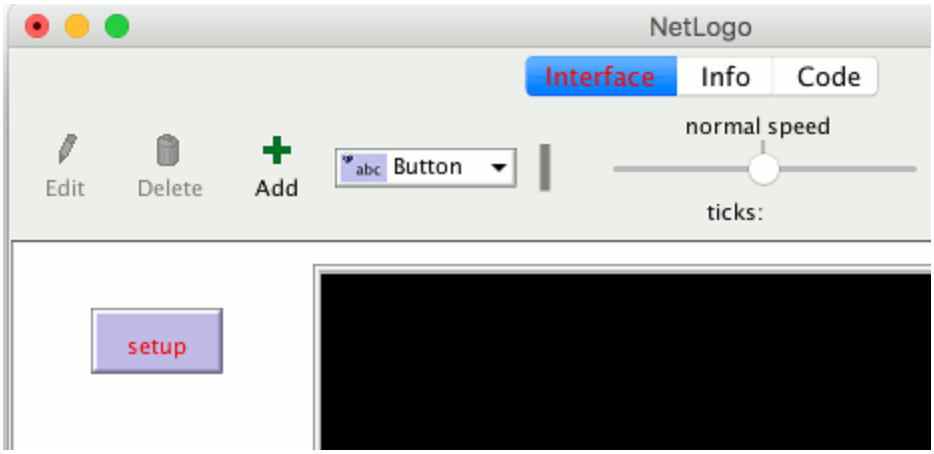
ایجاد دکمه setup

برای شروع یک مدل جدید، از منوی File گزینه "New" را نتخاب کنید. سپس با ایجاد دکمه "setup" فرایند ساخت مدل را شروع کنید.

- روی آیکن " add" موجود در نوار ابزار بالای تب واسط کلیک کنید.
- بر منوی مجاور add، گزینه button (دکمه) را انتخاب کنید (اگر قبلا انتخاب نشده باشد)
- یک جایی سفید رنگی از تب واسط کلیک کنید تا button ظاهر شود.
- یک جعبه محاورهای[66] برای ویرایش button باز می شود که شما در داخل قسمت " commands" می باید کلمه setup را تایپ کنید.
- وقتی کارتان تمام شد، دکمه OK را بزنید و جعبه محاوره را ببندید.

اکنون شما یک دکمه setup را ایجاد کرده اید. فشار دادن این دکمه باعث اجرای رویهای بنام "setup" می شود. یک رویه در واقع یک توالی از فرمانها (دستورات) NetLogo است که ما نام جدیدی را به آنها اختصاص میدهیم. ما به زودی این رویه را تعریف میکنیم والی الان آن را نداریم. دکمه setup به رویهای ارجاع می کند که وجود ندارد (تعریف نشده) از این رو رنگ دکمه قرمز میشود.

[66] Dialog box



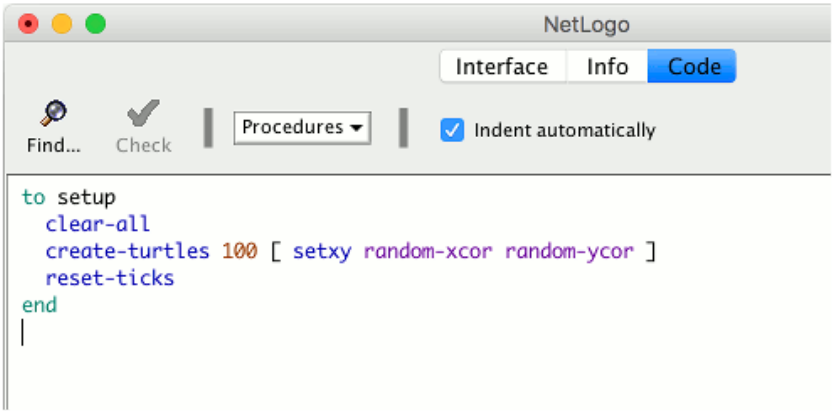
اگر شما می خواهید پیام واقعی خطا را مشاهده کنید روی دکمه کلیک کنید.

اکنون ما رویه "setup" را خواهیم ساخت و از این رو پیام خطا رفع خواهد شد.

- به تب کد بروید.
- عبارت زیر را تایپ کنید:

```
to setup
  clear-all
  create-turtles 100 [ setxy random-xcor random-ycor ]
  reset-ticks
end
```

وقتی کارتان تمام شد، تب کد به شکل زیر خواهد بود:



توجه داشته باشید که برخی از خطوط کد تورفتگی[67] دارند. برخی از افراد ترجیح میدهند کدهایشان دارای تو رفتگی باشد و اگر چه اینکار اجباری نیست ولی خوانایی و مدیریتپذیری کد را افزایش میدهد.

رویه شما با to آغاز و با end تمام می شود. تمام رویه ها با این کلمات آغاز و پایان مییابند.

بیایید به آنچه تایپ کردهاید نگاهی بیندازیم و ببینیم که هر خط رویه شما چه کاری انجام می دهد:

- to setup باعث تعریف کردن رویه setup میشود
- clear-allباعث بازگرداندن جهان (نمای مدل) به حالت اولیه و خالی میشود. تمام پَچها سیاه میشوند و تمام تِرتِلهایی که شما قبلاً ساختهاید ناپدید میشوند. اساساً این محیط را برای اجرای یک مدل جدید پاک میکند.
- create-turtles 100 منجر به ایجاد 100 تِرتِل میشود. آنها از نقطه مبدأ شکل میگیرند یعنی مرکز پَچ 0 0.
- بعد از create-turtles ما می توانیم فرمانی برای اجرا به تِرتِلهای جدید بدهیم که (این فرمانها) توسط براکت های مربعی [] محصور شده اند.
- setxy random-xcor random-ycor یک فرمانی است که از "ریپورترها"[68] استفاده کرده است. بر خلاف یک فرمان[69] یک ریپورتر یک نتیجه را گزارش میکند ابتدا هر تِرتِل ریپورتر random-xcor را اجرا می کند که (این ریپورتر) یک عدد تصادفی را از محدوده مجاز مختصات تِرتِل در امتداد X گزارش می دهد. سپس هر تِرتِل ریپورتر random-ycor را اجرا میکند که (این ریپورتر نیز) یک عدد تصادفی را از محدوده مجاز مختصات تِرتِل در امتداد Y

[67] Indent

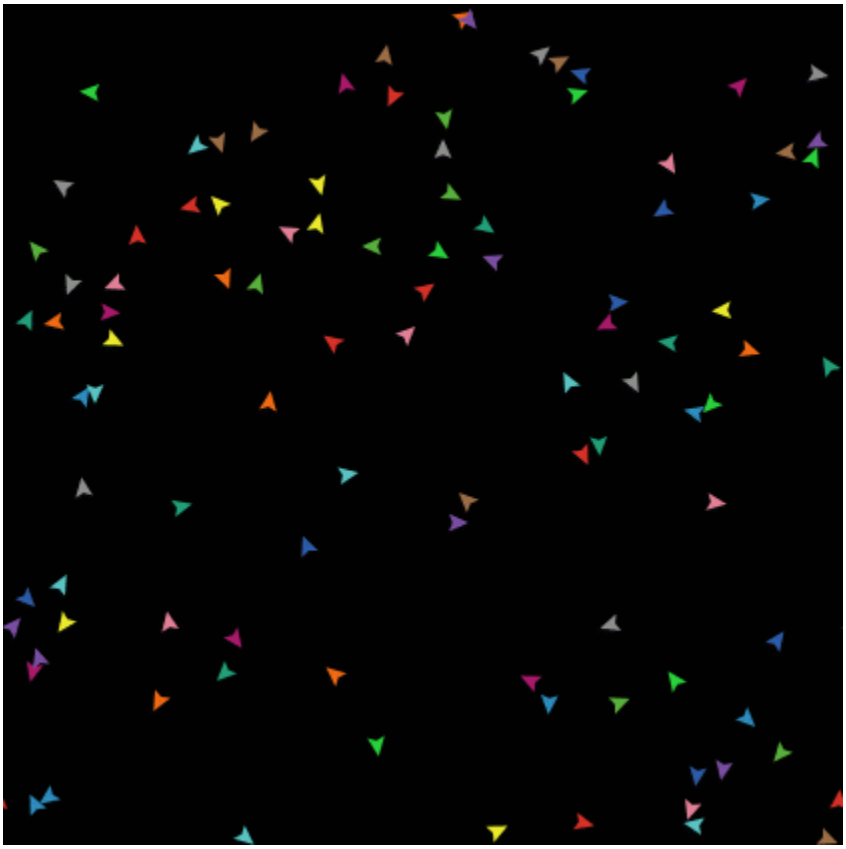
[68] در Netlogo ریپورترها به رنگ بنفش هستند مثل xcor یا pxcor (مترجم)

[69] در Netlogo فرمانها به رنگ آبی هستند مثل set یا ask (مترجم)

گزارش میدهد. سرانجام هر تِرتِل دستور setxy را با این دو عدد به عنوان ورودی اجرا می کند. این باعث میشود که تِرتِل به نقطه ای با این مختصات حرکت کند.

- reset-ticks شمارنده تیک را آغاز میکند. اکنون دیگر رویه setup کامل شده است
- end تعریف رویه "setup" را تکمیل میکند.

وقتی کارتان انجام شد، به تب واسط برگردید و دکمه setup را که پیشتر ساختهاید فشار دهید. شما خواهید دید که تِرتِلها در سراسرجهان پراکنده شدهاند:



چند دفعه پشت سرهم دکمه setup را فشار دهید. میبینید که هر دفعه چگونه ترتیب (جایگیری در محیط) تِرتِلها تغییر میکند. توجه داشته باشید که این امکان هم وجود دارد که برخی تِرتِلها درست روی هم بیافتند.

اندکی به این بیاندیشید که برای این که این اتفاق بیافتد شما لازم بود چه کارهایی انجام دهید. شما لازم بود که یک دکمه setup در تب واسط ایجاد کنید و یک رویه ای نیز در تب کد بسازید تا این دکمه از آن استفاده کند. فقط زمانی که شما این دو مرحله جداگانه را کامل انجام دهید، دکمه setup میتواند کار کند. در ادامه این آموزش، برای افزودن یک ویژگی دیگر به مدل، شما اغلب مجبور خواهید بود تا دو یا چند گام مشابه (با آنچه توضیح داده شد) را انجام دهید. سعی کنید چندین بار این مرحله را به صورت مستقل اجرا کنید تا دید

خوبی بگیرد و اگر رویهای ایجاد کردید که متوجه شدید کار نمیکند باز هم مطالب این بخش را مرور کنید تا متوجه شوید چه گامی را جا گذاشتهاید.

برگشت به به روزرسانی های مبتنی بر تیک

حال که ما داریم از شمارشگر تیک استفاده میکنیم ما باید به نتلوگو بگوییم که درعوض بهروزرسانی (آپدیتینگ) مداوم مدل، آن را یکبار در هر تیک بهروزرسانی نماید.

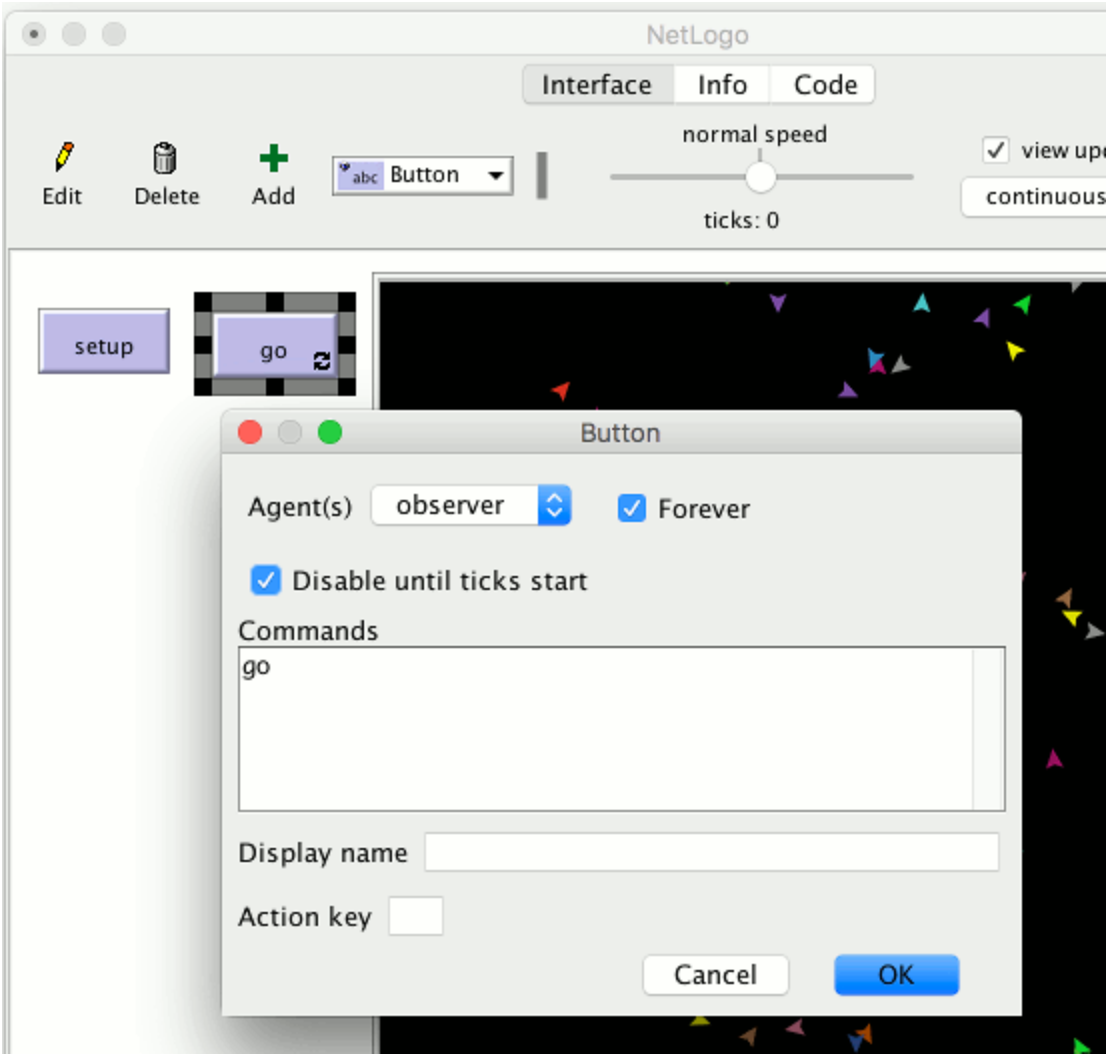
- منوی بهروزرسانی تیک را پیدا کنید. این منو در بالای نمای مدل قرار دارد و به صورت پیش فرض "continuous" میباشد.
- درعوض گزینه "on ticks" را انتخاب کنید.

این باعث می شود مدل شما سریعتر اجرا شود و ظاهر آن ثبات بهتری داشته باشد (از آنجا که به روزرسانی ها در زمان های ثابت اتفاق می افتد). برای بحث کامل در مورد به روزرسانی ها ، به **راهنمای برنامه نویسی** مراجعه کنید.

ایجاد دکمه go

اکنون دکمهای بنام "go" بسازید. همان گامهایی را که برای ساخت دکه "setup" انجام داید را دنبال کنید به جز موارد زیر:

- در قسمت commands بجای کلمه setup کلمه go را تایپ کنید.
- گزینه forever را در دیالوگ محاورهای فعال کنید
- گزینه Disable until ticks start را نیز فعال کنید.



گزینه " forever" باعث میشود وقتی دکمه یکبار فشار داده شد دیگر سرجای خود بماند و برنگردد. از این رو فرمانها در عوض یکبار، بارها و بارها تکرار میشوند.

گزینه " Disable until ticks start" شما را مجبور میکند تا ابتدا دکمه " setup " و سپس دکمه "go " را کلیک کنید.

- سپس رویه go را به تب کد اضافه کنید:

```
to go
  move-turtles
  tick
end
```

tick یک سینتَکس (از نوع فرمان) است که در هر تیک یک واحد شمارشگر تیک را به جلو میبرد.

اما move-turtles چیست؟ آیا یک سینتَکس است (بعبارتی در NetLogo از پیش تعریف شده است مانند فرمان و ریپورتر)؟ پاسخ به این سوال خیر است. این هم نوع دیگری از رویه است که شما قرار است به مدل اضافه کنید. تاکنون ما شما را با دو نوع از رویه یعنی setup و go که شما خودتان به مدل اضافه کردید، آشنا نمودیم.

- رویه move-turtles را بعد از رویه go اضافه کنید.

```
to go
  move-turtles
  tick
end

to move-turtles
  ask turtles [
    right random 360
    forward 1
  ]
end
```

توجه داشته باید که در move-turtles هیچ فاصلهای در اطراف خط تیره وجود ندارد. در آموزش شماره 2، به منظور تفریق دو عدد، ما با فاصله از 2 - red استفاده کردیم اما در اینجا بدون فاصله از move-turtles استفاده میکنیم. خط تیره "-" دو کلمه " move" و " turtles" را به یک اسم واحد تبدیل میکند.

در ادامه نقش هر فرمان در رویه move-turtles تشریح شده است:

- ask turtles [.....] میگوید که هر تِرتِل باید فرمانهای داخل براکت را انجام دهد.
- right random 360فرمان دیگری است که از یک ریپورتر استفاده می کند. در ابتدا هر تِرتِل یک عدد تصادفی حسابی را بین 0 تا 359 انتخاب میکند (ریپورتر random آن عددی که شما به آن دادهاید را برنمیگرداند و تا یک عدد قبل از آن را برمیگرداند). سپس، تِرتِل به اندازه این عدد به سمت راست میچرخد.
- Forward 1 باعث میشود که تِرتِل یک گام بجلو حرکت کند.

چرا ما بجای نوشتن این فرمانها در یک رویه جدا، آنها را در رویه go ننوشتیم؟ جواب این سوال این است که ما میتوانستیم که این کار را انجام دهیم ولی چون در خلال توسعه مدل، شما مجبورید بخشهای زیادی به مدل اضافه کنید، برای افزایش خوانایی کار ما آنها را در رویههای جداگانهای قرار میدهیم. ما مایلیم رویه

go تا جای ممکنه ساده بماند بطوریکه به سادگی قابل فهم باشد. با اضافه کردن این رویهها شما چیزهای زیادی را در مدل تعبیه میکنید که مدل در هنگام اجرا تمام آنها را انجام میدهد چیزهای نظیر محاسبه یک ورودی و ترسیم یک نتیجه در قالب یک نمودار. هر کدام از این چیزها در داخل یک رویه بخصوص واقع شده است که هر رویه نیز نامی منحصر به فرد دارد.

دکمه go که شما در تب واسط ساختید یک دکمه forever (تکرار) می بالشد یعنی با کلیک کردن روی آن مدل بصورت بدون وقفه اجرا میشود مگر تا زمانی که شما آن را متوقف کنید (دوباره روی آن کلیک کنید). وقتی که یکبار دکمه "setup" را فشار دادید و تِرتِلها را ایجاد کردید، دکمه "go" را فشار دهید و ببینید چه اتفاقی میافتد. آن را خاموش کنید (یعنی مجدد روی دکمه go کلیک کنید) مشاهده خواهید کرد که تمام تِرتِلها در مسیرخودشان متوقف شده اند.

توجه داشته باشید وقتی یک تِرتِل در لبه محیط (جهان) حرکت میکند از آن سوی محیط ظاهر می شود به دیگر سخن محیط امکان تاخوردن دارد (محیط NetLogo به صورت پیش فرض تا میخورد و شما امکان تغییر آن رادارید برای اطلاعات بیشتر به بخش **توپولوژی** در **راهنمای برنامهنویسی** مراجعه کنید)

آزمایش با فرمانها

پیشنهاد میدهیم که از دیگر فرمانهای تِرتِل استفاده کنید.

فرمانهای را در مرکز فرمان تایپ کنید. مثل (turtles > set color red) یا فرمانهایی را به رویههای go، setup یا move-turtles اضافه کنید.

توجه داشته باشید هنگامی که فرمانها را در مرکز فرمان تایپ میکنید، بسته به نوع عاملی که قرار است فرمان را اجرا کند، شما باید یکی از <turtles>, <patches>, links یا observer> که در منوی حاصله سمت چپ میباشند را انتخاب کنید. این کار درست مثل ask turtles یا ask patches است (برای زمانی که از عامل observer> استفاده میشود) با این فرق که دیگر نیازی به تایپ این عبارات را ندارد. همچنین می توانید از کلید tab برای تغییر انواع عامل استفاده کنید ، که برای شما راحت تر از استفاده از منو است.

لطفاً در مرکز فرمان عامل turtles> را انتخاب و عبارت pen-down را تایپ و اجرا کنید، سپس دکمه go را فشار دهید.

افزون بر این، در داخل رویه move-turtles ، شما می توانید عبارت right random 360 را بهright random 45 تغییر دهید.

به ساختار مدل یک نگاهی بندازید، می بینید که چقدر ساده است و نتایج آن هم واضح و هم قابل فهماند--- این یکی از توانمندی NetLogo است.

وقتی احساس کردی که به اندازه کافی مدل را آزمایش نمودهاید، حال شما آمادهاید که مدل خودتان را توسعه دهید.

پَچ ها و متغیرها

حال ما 100 تِرتِل داریم که بی هدف در محیط پرسه میزنند، و هیچ اطلاعاتی از محیط اطرافشان ندارند. ما میخواهیم یک شکل و شمایلی به این محیط بدهیم.

- به رویه setup برگردید و شما میتوانید آن را به صورت زیر بازنویسی کنید

```
to setup
  clear-all
  setup-patches
  setup-turtles
  reset-ticks
end
```

- تعریف جدید setup به دو رویه جدید اشاره میکند. برای تعریف setup-patches عبارت زیر را اضافه کنید:

```
to setup-patches
  ask patches [ set pcolor green ]
end
```

رویه setup-patches باعث میشود که هر پَچ با رنگ سبز اغاز شود (متغییر زنگ تِرتِل color می باشد و متغر رنگ پَچ pcolor می باشد)

رویه setup-turtles تنها بخشی است که هنوز تعریف نشده است.

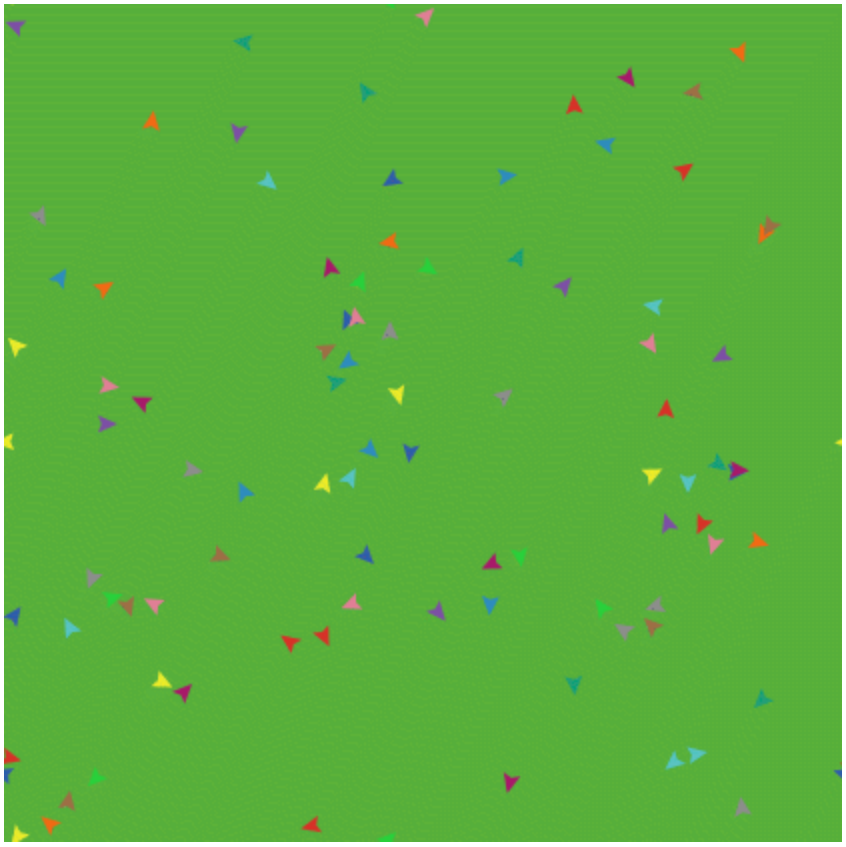
- برای اینکار، رویه زیر را اضافه کنید:

```
to setup-turtles
  create-turtles 100
  ask turtles [ setxy random-xcor random-ycor ]
end
```

به این نکته توجه کردید که رویه جدید setup-turtles حایز بسیاری از فرمانهای رویه قبلی setup میباشد؟

- دوباره به تب واسط برگردید.
- دکمه setup را فشار دهید.

چه جالب یک محیط زیبایی با پس زمینه سبز ظاهر شده که تِرتِلها در جای جای ان پراکندهاند.



حال که طرز کار رویه جدید setup را مشاهده کردید، بد هم نیست مجدداً بخشهای (رویهها) آن را بررسی کنید.

متغیرهای تِرتِل

در این مقطع ما تعدادی تِرتِل داریم که در یک محیط میگردند، اما هیچکاری با آن ندارند. بیایید یک ارتباطی بین تِرتِلها و محیط (یعنی پَچها) برقرار کنیم.

ما کاری میکنیم که تِرتِلها بتوانند علف بخورند (پَچهای سبز)، تولید مثل کنند و حتی بمیرند. (و) علفها بعد از آنکه خورده شدند با نرخ آهستهای دوباره رشد میکنند.

وقتی که یک تِرتِل تولید مثل میکند یا میمیرد ما به شیوهای از کنترل کردن نیاز داریم. ما این کار را از طریق رصد تغییرات "انرژی" تِرتِلها انجام میدهیم. برای اینکار ما باید یک متغیر جدید برای تِرتِلها تعریف کنیم.

ما تاکنون متغیرهای تِرتِلی از پیش تعریف شدهای مانند color را دیدهایم. برای تعریف یک متغیر تِرتِلی جدید می باید از کلید واژه turtles-own [....] در بالای تب کد استفاده کرد و متغیر energy را در آن قرار داد. توجه کنید که این باید بالای همه رویه ها (یعنی setup، go و .....) باشد.

```
turtles-own [energy]

to seup
  clear-all
  setup-patches
  setup-turtles
  reset-ticks
end

to go
  move-turtles
  tick
end
```

بیایید متغیر جدید (energy) را تعریف کنیم تا تِرتِلها قادر به خوردن باشند.

- به کد تب برگردید
- رویه go را به صورت زیر بازنویسی کنید

```
to go
  move-turtles
  eat-grass
  tick
end
```

- رویه جدید eat-grass را اضافه کنید.

```
to eat-grass
  ask turtles [
    if pcolor = green [
      set pcolor black
      set energy energy + 10
    ]
  ]
end
```

ما داریم از فرمان if برای اولین بار استفاده میکنیم. بادقت به کد نگاه کنید. هر تِرتِل وقتی فرمانها را اجرا میکند ابتدا رنگ پَچی (pcolor) که روی آن قرار دارد را بررسی میکند که ببیند سبز است یا خیر (توجه کنید تِرتِلها به متغیرهای پَچهایی که روی آنها واقع هستند دسترسی متقسیم دارند) اگر رنگ پَچ سبز بود به این معناست که قسمت شرط برقرار (یعنی درست) است و تِرتِل تمام فرمین داخل براکت را انجام میدهد در غیر اینصورت کاری انجام نمی دهد. این فرمانها باعث میشوند که تِرتِل رنگ پَچ زیرپای خود را سیاه کند و انرژی خود را 10 واحد افزایش دهد. هنگامی که پَچ سیاه میشود به این معناست که علف آن خورده شده است و انرژی تِرتِلی که آن را خورده 10 واحد افزایش مییابد.

درگام بعد، بگذارید کاری بکنیم که حرکت کردن تِرتِل باعث مصرف میزانی از انرژی آن شود.

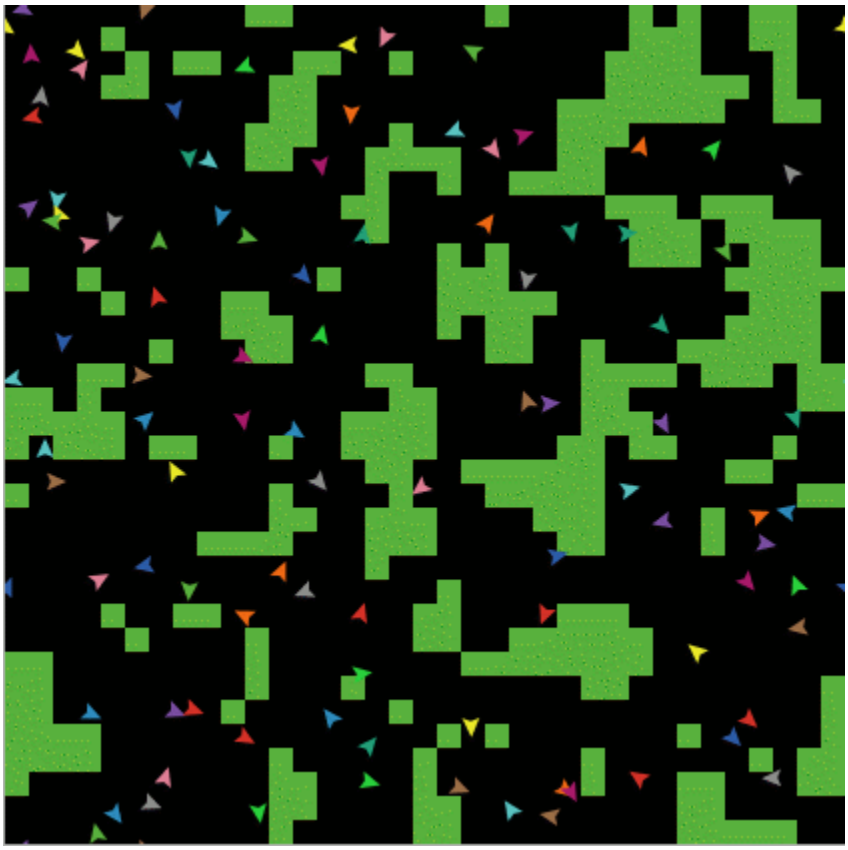
- رویه turtle-moves را به صورت زیر بازنویسی کنید:

```
to move-turtles
  ask turtles [
    right random 360
    forward 1
    set energy energy - 1
  ]
end
```

به موازاتی که هر تِرتِل در محیط پرسه میزند، آن تِرتِل در هر گام یک واحد انرژی از دست میدهد.

- حال یه تب واسط برگردید و ابتدا دکمه setup و سپس دکمه go را فشار دهید.

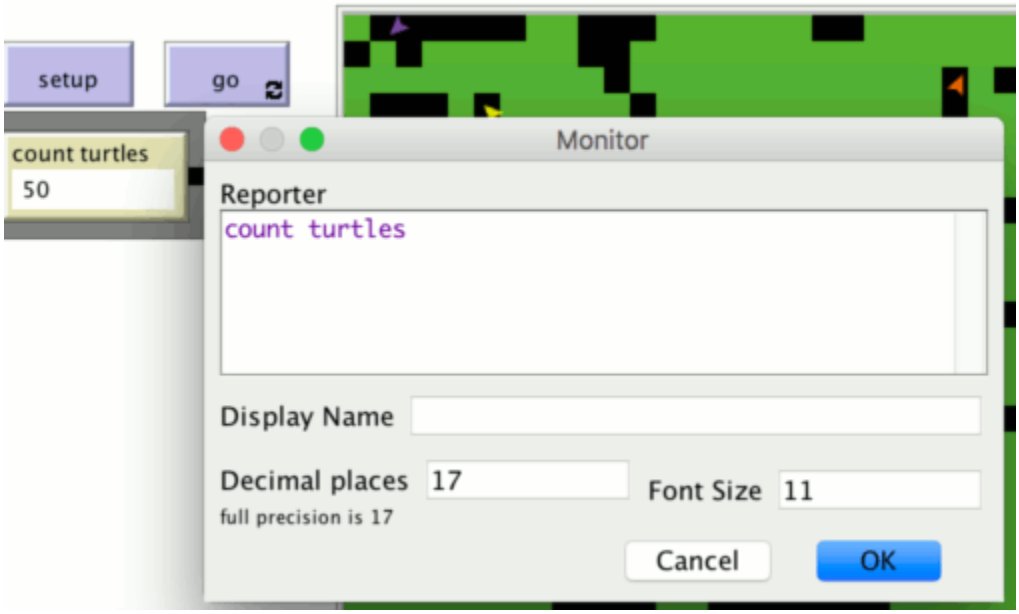
خواهید دید که هنگامی که تِرتِلها روی پَچها حرکت میکنند، رنگ آنها سیاه میشود.



مانیتورها

سپس با ستفاده از نوارابزار، شما اقدام به ساختن دو مانیتور مینمایید (با استفاده از آیکن add در نوارابزار، شما مثل حالتی که دکمه ها و اسلایدرها را ساختید این کار را انجام دهید)

- روی آیکن " add" موجود در نوار ابزار بالای تب واسط کلیک کنید بر منوی مجاور add، گزینه Monitor (مانیتور) را انتخاب کنید. سپس در محیط واسط یک کلیک بنمایید.
  یک جعبه محاورهای باز می شود
- در جعبه محاورهای عبارت: count turtles را تایپ کنید (طبق تصویر زیر)
- با فشار دادن دکمه OK، جعبه را ببندید.



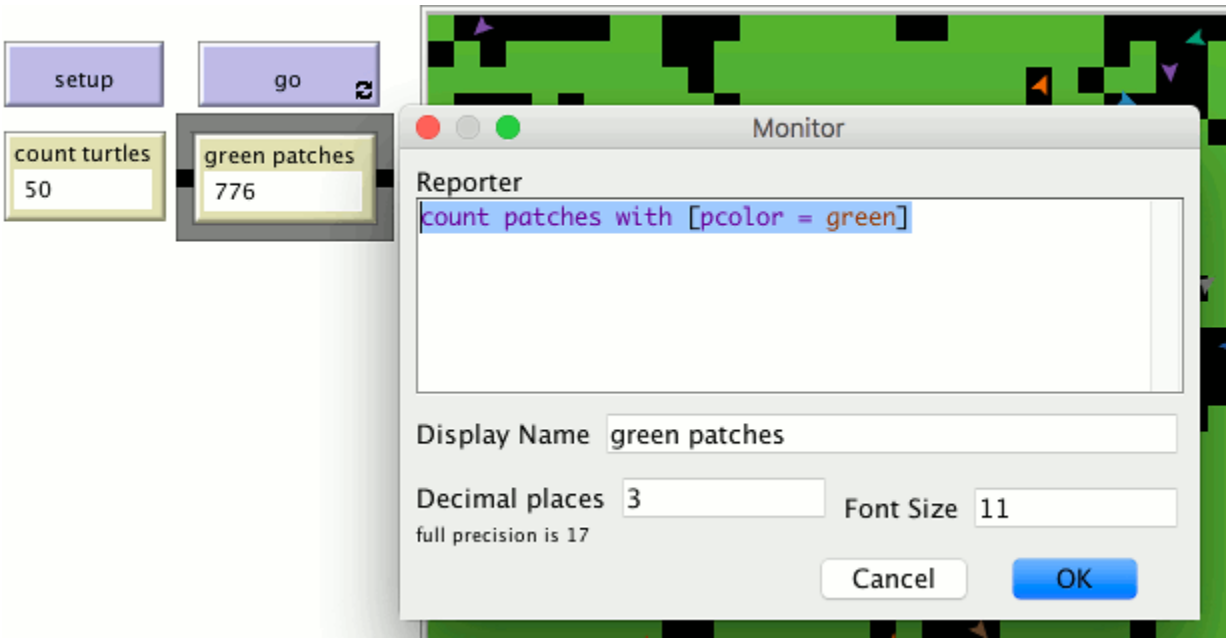
turtles یک مجموعه عامل[70] است، یعنی مجموعه ای از تمام تِرتِلها. count به ما میگوید که چه تعداد تِرتِل در این مجموعه وجود دارد.

حال بیایید مانیتور دوم را ایجاد کنیم

- روی آیکن " add" موجود در نوار ابزار بالای تب واسط کلیک کنید بر منوی مجاور add، گزینه Monitor (مانیتور) را انتخاب کنید. سپس در محیط واسط یک کلیک بنمایید.
  یک جعبه محاورهای باز می شود

[70] agentset

- در جعبه محاورهای عبارت: [count patches with [pcolor = green را تایپ کنید (طبق تصویر زیر)
- در قسمت Display name عبارت green patchesرا تایپ کنید.
- با فشار دادن دکمه OK، جعبه را ببندید.



در اینجا ما دوباره از count برای شمارش اعضای یک مجموعه عامل استفاده میکنیم. patches مجموعه تمام پَچهاست ولی ما نمیخواهیم بدانیم که پَچها چه تعداد می باشند بلکه میخواهیم بدانیم چه تعداد پَچ سبز داریم. این کاری است که with انجام میدهد. این باعث انتخاب مجموعه کوچکتری از عاملها میشود که شرط داخل براکت برای آنها برقرار (یعنی درست) است. شرط شامل pcolor = green میباشد بطوریکه این فقط تعداد پَچهای سبز را برمیگرداند.

اکنون دو مانیتور داریم که تعداد تِرتِلها و پَچها را گزارش می دهند تا به ما کمک کنند آنچه در مدل ما میگذرد را رصد کنیم. با اجرای مدل، اعداد در مانیتورها به طور خودکار تغییر میکنند.

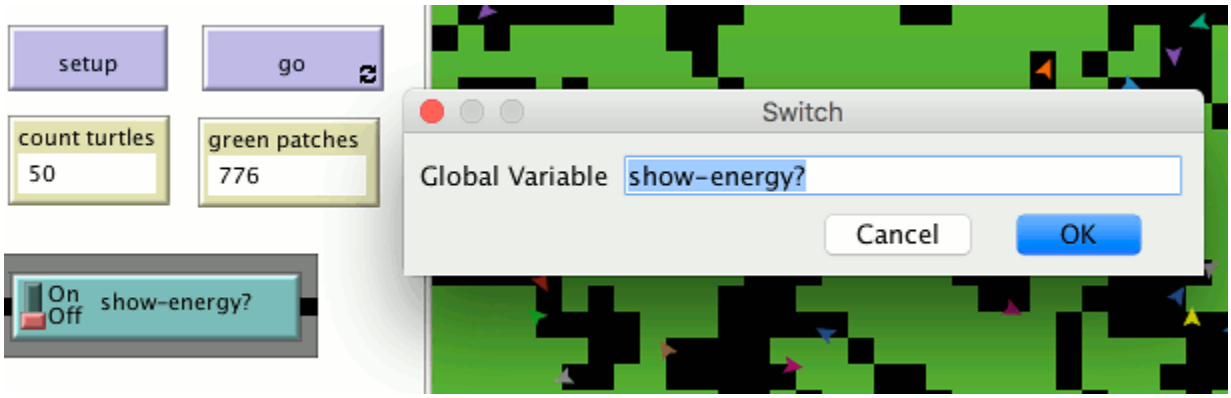
- از دکمه setup و go استفاده کنید و سپس تغییرات عددی مانیتورها را مشاهده کنید.

سوییچ ها و برچسب ها

کار تِرتِلها فقط سیاهکردن پَچها نیست (یعنی غذا خوردن) بلکه آنها انرژی بدست میآورند و انرژی هم از دست میدهند. هنگامی که مدل اجرا می شود، از یک مانیتور تِرتِل استفاده کنید تا فرایند افزایش یا کاهش انرژی آن را مشاهده کنید.

زیباتر خواهد بود اگر ما می توانستیم انرژی تک تک تِرتِلها در یک لحظه مشاهده کنیم. اکنون ما دقیقاً این کار را انجام میدهیم و با افزودن یک سوییچ به مدل، امکان روشن یا خاموش کردن اطلاعات بصری بیشتری را فراهم میکنیم.

- روی آیکن " add" موجود در نوار ابزار بالای تب واسط کلیک کنید بر منوی مجاور add، گزینه Switch (مانیتور) را انتخاب کنید. سپس در محیط واسط یک کلیک بنمایید.
  یک جعبه محاورهای باز می شود
- در فیلد Global variable عبارت show-energy? را تایپ کنید علامت سوال انتهای آن فراموش نشود (مطابق شکل زیر).
-



- اکنون با استفاده از تب کد نوارافزار، به رویه "go" برگردید.
- رویه eat-grass را به شرح زیر بازنویسی کنید:

```
to eat-grass
  ask turtles [
    if pcolor = green [
      set pcolor black
      set energy energy + 10
    ]
    ifelse show-energy?
      [ set label energy ]
      [ set label "" ]
```

```
  ]
end
```

رویه eat-grass? فرمان ifelse را معرفی میکند. با دقت به کد نگاه کنید. هر تِرتِل هنگامی که فرمانهای جدید را اجرا میکند ابتدا مقدار show-energy? را چک میکند (که توسط سوییچ تعیین میشود) اگر سوییچ روشن (یعنی on) بود نتیجهی مقایسه (یعنی شرط) برقرار است لذا تِرتِل فرمانهای موجود در براکت اول را اجرا میکند که در این حالت مقدار انرژی خود را در قالب برچسب خود نشان میدهد ولی اگر شرط برقرار نبود (یعنی سوییچ خاموش (off) باشد) تِرتِل فرمانهای موجود در براکت دوم را اجرا می کند یعنی در این حالت متن موجود در برچسب را حذف میکند (آن را معادل با هیچ چیز میگذارد)

(در NetLogo، قطعهای از متن "رشته" نامیده میشود، مخفف رشتهای از کارکترها. یک رشته توالی از حروف و سایر کارکترهاست که بین دو نقل قول نوشته میشوند. در اینجا دو نقل[71] قول داریم که بین آنها هیچ چیز نیست یعنی "" این یعنی یک رشته خالی. اگر برچسب یک تِرتِل، دو نقل قول خالی باشد هیچ متنی به آن تِرتِل پیوست نمیشود)

- حال عملکرد سوییچ را در مدل آزمایش کنید. برای اینکار ابتدا دکمه سsetup و سپس go را فشار داده و در حین اجرای مدل چندباری سوییچ را خاموش و روشن کنید.

هنگامی که سوییچ روشن است. شما می بینید که هر وقت تِرتِل مقداری علف می خورد انرژی آن بالا می رود و همچنین میبینید وقتی آن حرکت میکند از انرژیاش کم میشود.

رویه های بیشتر

حال تِرتِلهای ما توان غذل خوردن دارند. بیایید امکان تولید مثل و مردن را به آنها نیز بدهیم. افزون بر این کاری بکنیم که علف دوباره رشد کند. ما هم اکنون از طریق سه رویه مختلف یعنی هر رویه برای یک رفتار، این سه رفتار جدید را به مدل اضافه میکنیم،

- به تب کد بروید.
- رویه go را به شرح زیر بازنویسی کنید.

```
to go
  move-turtles
  eat-grass
  reproduce
  check-death
  regrow-grass
  tick
```

[71] Double quotes

```
end
```

- به صورت زیر رویههای check-death ، reproduce و regrowth-grass را اضافه کنید.

```
to reproduce
  ask turtles [
    if energy > 50 [
      set energy energy - 50
      hatch 1 [ set energy 50 ]
    ]
  ]
end

to check-death
  ask turtles [
    if energy <= 0 [ die ]
  ]
end

to regrow-grass
  ask patches [
    if random 100 < 3 [ set pcolor green ]
  ]
end
```

هر یک از این رویهها از فرمان if استفاده میکند. در هنگام اجرای رویه check-grass، هر تِرتِل این نکته را چک می کند که اگر میزان انرژیاش مساوی یا کمتر از عدد صفر بود (یعنی شرط برقرار باشد)، بمیرد (die یکی از سینتَکسهای از پیش تعریف شده در NetLogo است).

زمانیکه هر تِرتِل رویه reproduce را اجرا میکند. آن تِرتِل مقدار متغیر انرژی (energy) خود را بررسی میکند. اگر مقدار انرژی بزرگتر از 50 بود، سپس تِرتِل فرامین داخل اولین براکت را انجام میدهد. در این حالت، او مقدار انرژی را تا 50 واحد کاهش میدهد سپس یک تِرتِل جدیدی با 50 واحد انرژی به دنیا میآورد. hatch فرمان یک سینتکس از پیش ساخته شده است که به این صورت است hatch number [commands] . وقتی تِرتِلی این فرمان را اجرا میکند به اندازه number تِرتِل جدید بدنیا میآورد که کاملاً همسان خودش میباشند و از آنها میخواهد که فرامین داخل براکت [commands] را اجرا کنند. شما میتوانید از فرمانهای مختلفی استفاده کنید بعنوان مثال از تِرتِلهای جدید بخواهید رنگ یا جهت خود را تغییر دهند و یا هرکاری دیگری که میخواهید انجام دهند. در حالت فعلی ما فقط یک فرمان را مینویسیم و مقدار انرژی تِرتِلهای جدید را معادل 50 قرار میدهیم (و آنها این را اجرا میکنند).

زمانیکه هر پَچ رویه regrowth-grass را اجرا میکند. آن پَچ بررسی میکند که اگر در بازهی اعداد تصادفی صحیح 0 تا 99، عدد تصادفی کوچکتر از 3 ایجاد شود، رنگ خود را سبز میکند. این به طور متوسط در 3% مواقع برای پَچها اتفاق میافتد چون از بین 100 حالت ممکنه فقط سه عدد صیح (0، 1 و 2) کوچکتر از 3 میباشند.

- حال به تب واسط برگردید و دکمه setup و سپس go را فشار دهید.

اکنون شما رفتارهای جالبی در مدل میبینید. برخی تِرتِلها میمیرند، تعدادی تِرتِل جدید متولد میشود (بدنیا میآید) و بعضی پَچها نیز دوباره رشد میکنند. این دقیقاً چیزی است که ما میخواستیم انجام دهیم.

اگر مانیتورهای مدل را نگاه کنید میبیند که count turtles و green patches هر دو در نوسان هستند. آیا این الگوی نوسان قابلپیش بینی است؟ آیا رابطهای بین متغیرها وجود دارد؟

خوب است اگر راه آسانتری برای رصد (ردیابی) تغییرات رفتار مدل در طول زمان داشته باشیم. NetLogo به ما اجازه میدهد تا در ادامهی کار دادهها را رسم کنیم. این مرحله بعدی ما خواهد بود.

ترسیم نمودار

برای استفاده از نمودار لازم است تا آن را در تب واسط ایجاد کرده و تعدادی دستور در داخل آن قرار دهیم.

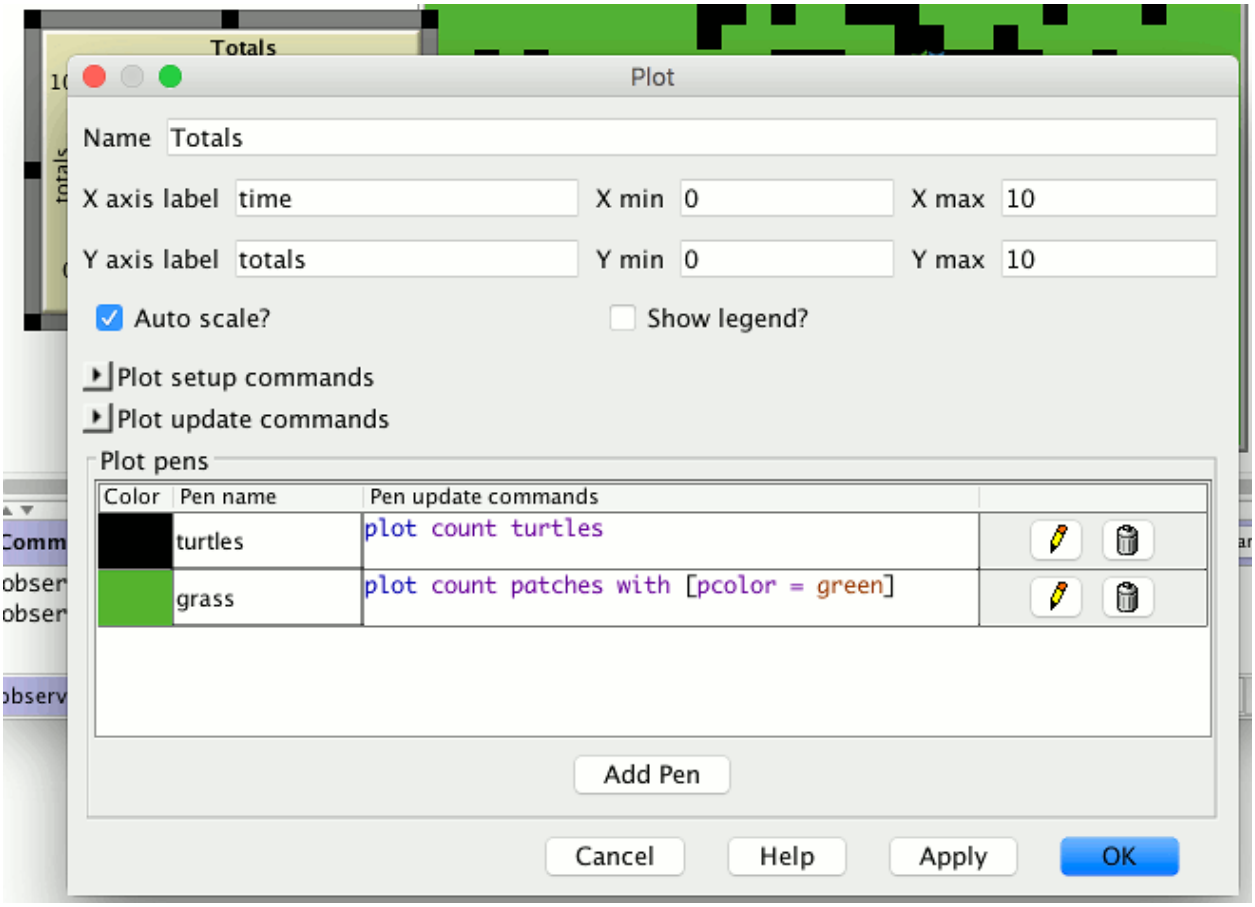
هنگامی که رویه setup فرمان reset-ticks و رویه go فرمان tick را فراخوانی میکنند، فرمانهایی که ما در نمودارها قرار دادهایم بصورت خودکار اجرا میشوند.

- روی آیکن " add" موجود در نوار ابزار بالای تب واسط کلیک کنید بر منوی مجاور add، گزینه Plot (نمودار) را انتخاب کنید. سپس در محیط واسط یک کلیک بنمایید.

یک جعبه محاورهای باز می شود

- در فیلد Name کلمه "Totals" را بنویسید. (به شکل زیر نگاه کنید)
- عنوان محور X را بگذارید "time"
- عنوان محور Y را بگذارید "totals"
- در پایین Pen name بنویسید "turtles"
- درپایین Pen update commnds بنویسید plot count turtles
- دکمه "add pen" را فشار دهید (یک قلم دیگر ایجاد میشود)
- در پایین Pen name بنویسید "grass"
- درپایین Pen update commnds بنویسید plot count patches with [pcolor = [green]
-

وقتی کارتان تمام شد، دیالوگ محاورهای به صورت زیر خواهد بود.



- برای اتمام ویرایش دکمه Ok را فشار دهید.

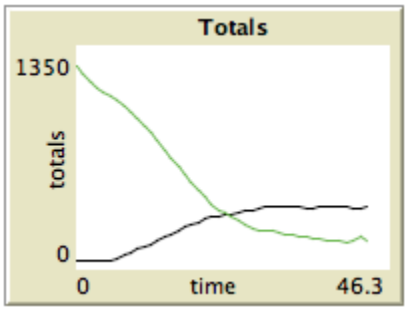
توجه داشته باشید وقتی شما نمودار را میسازید میتوانید مقادیر حداقلی و حداکثری محورهای X و Y را نیز مشخص کنید. با فعال گذاشتن تیک گزینه مقیاسدهی خودکار (Auto Scale?)، به نمودار این امکان را میدهسد که اگر مقادیر محورهای X و Y از حد مورد نظر عبور کردند، محورهای نمودار به صورت خودکار افزایش مقیاس یابند و داده را نشان دهند.

توجه داشته باشید که فرمان plot که در پایین pen update commnds نوشته می شود یک ورودی می گیرد (در حالت اول تعداد کل تِرتِلها و در حالت دوم تعداد کل پَچهای سبز) که این ورودی در محور Y رسم میشود و در هر تیک با تغییر محور زمانی X، مقدار Y نیز تغییر میکند.

- حال به تب واسط برگردید و دکمه setup و سپس go را فشار دهید.

مشاهده میکنید که به موازات اجرای مدل، نمودار نیز رسم میشود. مدل شما شکل عمومی زیر را خواهد داشت اگر چه شاید دقیقاً مشابه نباشند (به این علت که شاید شما از سیستم عامل دیگری استفاده کنید)

بیاد آوریدکه ما تیک گزینه مقیاسدهی خودکار (Auto Scale?) را فعال گذاشته بودیم. بنابراین وقتی محورهای X و Y از حد مورد نظر خودشان عبور کردند، مقیاس محورهای نمودار بصورت خودکار افزایش مییابد.



اگر اسم قلمها (نمودارها) را فراموش کردهاید میتوانید نمودار را ویرایش نموده و تیک گزینه " Show legend?" را فعال کنید.

شما ممکن است به دفعات مختلف مدل را اجرا کنید و ببینید چه چنبههایی از مدل تغییر نمیکند و چه جنبههایی هم در هر اجرا تغییر میکند.

شمارنده تیک

برای مقایسه عملکرد مدل در اجراهای مختلف، شایسته است که این کار برای یک بازه زمانی مشخص انجام دهید.(در واقع) آگاهی از اینکه چگونه در یک زمان مشخص، یک اقدام را اجرا و یا متوقف کنید موضوع مهمی است که فقط زمانی حاصل می شود که شما بفهمید چند بار رویه go اجرا شده است. دقیقاً با فهم این نکته شما میتوانید تحلیلهای مناسبی از رفتار مدل در بازههای زمانی مختلف داشته باشید. این کاری است که شمارنده تیک انجام میدهد.

شما قبلاً با بهرهگیری از فرمانهای reset-ticks و tick (که باعث فعالسازی مدل نیز میشوند) از شمارنده تیک استفاده کردهاید.

شما همچنین میتوانید از شمارنده تیک برای مقاصد دیگر نیز استفاده کنید. بعنوان نمونه ایجاد محدودیت زمانی برای اجرای مدل.

- رویه go را به صورت زیر تغییر دهید.

```
to go
  if ticks >= 500 [ stop ]
  move-turtles
  eat-grass
  check-death
  reproduce
  regrow-grass
  tick
end
```

- حال دکمه setup و سپس go را فشار دهید.

همانگونه که میبینید بعد از گذشت 500 تیک زمانی مدل به صورت خودکار متوقف می شود (یعنی تمام نمودارها و مانیتورها نیز متوقف می شوند)

فرمان tick شمارنده تیک را یک واحد جلو میبرد. و ticks ریپورتری است که مقدار فعلی شمارنده تیک را گزارش (ریپورت) میدهد. در رویه setup، توجه داشته باشید که وقتی این رویه اجرا می شود مقدار ریپورتر ticks روی عدد صفر تنظیم میشود.

برخی جزییات بیشتر

ابتدا بجای اینکه ما 100 تِرتِل داشته باشیم میتوان شمار متغیری از تِرتِلها را داشت.

- روی آیکن " add" موجود در نوار ابزار بالای تب واسط کلیک کنید بر منوی مجاور add، گزینه slider (اسلایدر) را انتخاب کنید. سپس در محیط واسط یک کلیک بنمایید.

یک جعبه محاورهای باز می شود

- در فیلد Global variable کلمه "number" را بنویسید.
- حداقل و حداکثر مقادیر در اسلایدر را (مطابق میلتان) تغییر دهید.
- سپس در داخل رویه setup-turtles بجای create-turtles 100 عبارت زیر را بنویسید

```
to setup-turtles
  create-turtles number [ setxy random-xcor random-ycor ]
end
```

این را آزمایش کنید و تاثیرات تعداد مختلفی از تِرتِلها (مثلا 50 تِرتِل، 150 تِرتِل و ....) را در گذر زمان در نمودار ها مشاهده کنید.

دوم، بد نیست امکان تعدیل مقدار انرژی تِرتِلها را نیز فراهم سازیم؟ (انرژی که از غذا خوردن بدست میآورند آید و از تولید مثل از دست میدهند)

- یک اسلایدری بنام energy-from-grass ایجاد کنید
- یک اسلایدر دیگر نیز بنام birth-energy ایجاد کنید.
- سپس در داخل eat-grass تغیر زیر را انجام دهید

```
to eat-grass
  ask turtles [
    if pcolor = green [
      set pcolor black
      set energy (energy + energy-from-grass)
    ]
    ifelse show-energy?
      [ set label energy ]
      [ set label "" ]
  ]
end
```

- و در داخل reproduce ، تغییر زیر را اعمال کنید.

```
to reproduce
  ask turtles [
    if energy > birth-energy [
      set energy energy - birth-energy
      hatch 1 [ set energy birth-energy ]
    ]
  ]
end
```

در نهایت آیا میتوانید تغییرات دیگری در مدل ایجاد کنید بعنوان نمونه آیا میتوانید یک اسلایدری برای تعدیل میزان رشد مجدد علفها ایجاد کنید؟ ایا میتوانید قواعد حرکتی برای تِرتِلها یا حتی تِرتِلهای جدید ایجاد کنید؟

آنچه در ادامه خواهد آمد

اکنون شما یک مدل ساده از یک اکوسیستم دارید. پَچها امکان رشد علف دارند، تِرتِلها میتوانند پرسه بزنند، عاف بخورند، تولید مثل کنند و حتی بمیرند.

شما یک واسطی از عناصر مختلفی چون دکمهها، اسلایدرها، سوییچها، مانیتورها و نمودار ساختهاید. شما حتی یکسری رویه نوشتهاید که به تِرتِلها میگویند چه کارهایی انجام دهند.

این آموزش در اینجا به پایان میرسد.

اگر مایلید در خصوص اجرای محیط واسط NetLogo اطلاعات مفصلی کسب کنید. بخش **راهنمای واسط** اطلاعات مفیدی به شما ارایه میدهد (منوها و تبها). اگر علاقمندید که جزییات بسیار بیشتری در خصوص چگونگی نوشتن رویهها بدانید میباید به بخش **راهنمای برنامهنویسی** مراجعه کنید. تمامی سینتَکسها نیز در **دیکشنری نتلوگو** فهرست و تسریح شده اند. با مطالعه این دیکشنری شما میتوانید نحوه استفاده از انواع سینتَکسها را به خوبی فرا گیرد.

شما حتی میتوانید مدلی که در این آموزش توسعه دادهاید را غنیسازی کنید. یعنی رفتارهای جدیدی به عاملهای آن اضافه کنید و یا حتی عاملهای دیگری به ان اضافه کنید.

متناوباً ، ممکن است شما بخواهید اولین مدل آموزشی یعنی Wolf Sheep Predation را دوباره مرور کنید. این مدلی است که شما در آموزش شماره 1 استفاده کرده اید. در مدل Wolf Sheep Predation ، گوسفندان را دیدید که در حال جابجایی هستند، منابعی (چمن) را مصرف میکنند که گهگاه آن ها دوباره رشد میکنند، آنها تحت شرایط خاصی تولید مثل میکنند و اگر منابعشان تمام شود میمیرند. اما این مدل گونه دیگری از موجودات در حال حرکت را نیز داشت، یعنی گرگ ها. افزودن گرگ ها به مدل مستلزم رویهها و سینتَکسهای جدیدی است. گرگ و گوسفند دو "گونه[72]" متفاوتی از تِرتِل هستند. برای دیدن نحوه استفاده از گونهها مجدداً مدل Wolf Sheep Predation را مطالعه کنید.

امیدواریم تا الان نکات کابردی را در خصوص زبان نتلوگو و چگونگی برنامهنویسی با آن یادگرفته باشید. در ادامه تمام رویههای مدل یکجا ارایه شده است.

ضمیمه: کد کامل

این مدل به صورت کامل با نام "Tutorial 3" در بخش Code Examples واقع در Model's Library قابل دسترس است.

[72] breed

توجه کرده اید که فهرست زیر سرشار از کامنتهایی (نظراتی) است که با علامت ویرگول (;) آغاز شده اند این کامنتها به شما امکان می دهند تا مستقیماً در سمت راست هر کد، توضیحاتی را درخصوص آن بنویسید. کامنتها به شدت میتوانند به خوانایی و قابل فهم بودن مدل توسط شما و دیگران کمک کنند.

در تب کد، رنگ کامنتها خاکستری است به طوریکه چشمان شما فوری میتوانند آنها را بخوانند.

```
turtles-own [energy] ;; for keeping track of when the turtle is ready
                     ;; to reproduce and when it will die

to setup
  clear-all
  setup-patches
  setup-turtles
  reset-ticks
end

to setup-patches
  ask patches [ set pcolor green ]
end

to setup-turtles
  create-turtles number    ;; uses the value of the number slider to create
turtles
  ask turtles [ setxy random-xcor random-ycor ]
end

to go
  if ticks >= 500 [ stop ]  ;; stop after 500 ticks
  move-turtles
  eat-grass
  check-death
  reproduce
  regrow-grass
  tick                    ;; increase the tick counter by 1 each time
through
end

to move-turtles
  ask turtles [
    right random 360
    forward 1
    set energy energy - 1  ;; when the turtle moves it looses one unit of
energy
  ]
end

to eat-grass
  ask turtles [
    if pcolor = green [
      set pcolor black
           ;; the value of energy-from-grass slider is added to energy
      set energy energy + energy-from-grass
    ]
  ifelse show-energy?
```

```
    [ set label energy ] ;; the label is set to be the value of the energy
    [ set label "" ]     ;; the label is set to an empty text value
  ]
end

to reproduce
  ask turtles [
    if energy > birth-energy [
      set energy energy - birth-energy  ;; take away birth-energy to give
birth
      hatch 1 [ set energy birth-energy ] ;; give this birth-energy to the
offspring
    ]
  ]
end

to check-death
  ask turtles [
    if energy <= 0 [ die ] ;; removes the turtle if it has no energy left
  ]
end

to regrow-grass
  ask patches [ ;; 3 out of 100 times, the patch color is set to green
    if random 100 < 3 [ set pcolor green ]
  ]
end
```