S1. شبیه‌سازی تفکیک نژادی.

در یک شهر، این که چرا افراد از نژادها یا فرهنگ‌های مختلف، محله‌های جداگانه‌ای برای سکونت خود تشکیل می‌دهند پرسشی است که پاسخ به آن کلید فهم بسیاری از دیگر پدیده‌های اجتماعی است. شبیه‌سازی با یک مدل ساده نشان می‌دهد که تنها یک عامل بسیار ساده و منطقی در تصمیم‌گیری خانواده‌ها برای انتخاب محل سکونت در کنار اثر نسبت زمین‌های آزاد به جمعیت منجر به شکل‌گیری این پدیده در سطح کلان می‌شود. فرض کنید هر خانواده تنها وقتی تصمیم می‌گیرد به محله‌ی دیگری نقل مکان کند که تعداد همسایگان هم‌نژاد با او از ۳۰ درصد کمتر باشند. در ظاهر چنین جامعه‌ای بسیار روادار و با ثبات تلقی می‌شود. نتیجه‌ای که از اجرای برنامه‌ی شبیه‌سازی بدست می‌آید با آنچه در واقعیت روی می‌دهد همخوانی بیشتری دارد.

برای شبیه‌سازی، شهر را به صورت یک جدول مربعی n×n در نظر بگیرید که در آن هر خانه نماینده‌ی یک خانه‌ی شهر است و فرض کنید فقط دو نژاد مختلف در این شهر زندگی می‌کنند. بدین ترتیب هر خانه یا خالی است یا با خانواده‌ای از نژاد ۱ و یا با خانواده‌ای از نژاد ۲ اشغال شده است. اگر حداقل s همسایه از ۸ خانه‌ی مجاور یک خانه با آن شبیه بودند آن خانواده همان جا می‌ماند در غیر اینصورت به نزدیکترین خانه‌ی خالی اطراف نقل مکان می‌کند.

در شروع هر یک از خانه‌های شهر را به یکی از نژادها به طور تصادفی نسبت دهید ولی در مجموع تنها d درصد خانه‌ها را پر کنید. (اصولا باید d<1 باشد تا تعدادی زمین آزاد برای تحرک وجود داشته باشد). شبیه‌سازی را تا $۱۰۰\ n^{۲}$ تکرار اجرا کنید و هر ۱۰۰ تکرار یکبار وضعیت شهر را با رنگ‌آمیزی خانه‌ها با سه رنگ مختلف نمایش دهید. فرض کنید خانواده‌ای که قصد نقل مکان دارد برای یافتن زمین آزاد، بلوک‌های مربع‌شکل به مرکز خانه‌ی فعلی خود را به ترتیب از کوچکترین عرض (۳) تا بلوکی به عرض ۲۰ خانه جستجو می‌کند و اگر زمین خالی پیدا نکرد به ناچار در مکان فعلی خود باقی می‌ماند (مشابه طرح زیر).

5	5	5	5	5	5	5
5	4	4	4	4	4	5
5	4	3	3	3	4	5
5	4	3	x	3	4	5
5	4	3	3	3	4	5
5	4	4	4	4	4	5
5	5	5	5	5	5	5

برنامه را به ازای s=1,2,3,4,5,6,7 و مقادیر مختلف d<1 و d>۰.۷ اجرا کنید. n را در حدود ۳۰ تا ۵۰ فرض کنید. برای نمونه به سایت http://ccl.northwestern.edu/netlogo/models/Segregation مراجعه کنید.

S2. شبیه‌سازی رأی‌گیری (Voting) در ساختار شبکه‌ای.

فرض کنید n^2 فرد در یک زمین مربعی $n \times n$ در کنار یکدیگر ایستاده‌اند. به طور دقیق‌تر فرض کنید هر فرد یک خانه از شبکه‌ی مربعی $n \times n$ را اشغال کرده است. هر فرد یک صفحه‌ی آبی‌رنگ و یک صفحه‌ی قرمز رنگ در دست دارد و هر بار که یک رأی‌گیری عمومی درخواست شود، فرد با بلند کردن رنگ مورد نظر رأی خود را اعلام می‌کند. در ابتدا فرض کنید هر فرد به طور کاملا تصادفی طرفدار رنگ آبی یا رنگ قرمز است. هدف از این شبیه‌سازی روشن کردن این پرسش است که اگر افراد در بیان رأی خود به رأی اطرافیان نگاه کنند و تحت تاثیر رأی اکثریت اطراف رأی بدهند چه نتیجه‌ای به بار خواهد آمد. فرض کنید رأی‌گیری‌های عمومی پشت سر هم انجام می‌شوند و در هر بار، فرد رأی خود را به رأیی که بیش از ۴ نفر از ۸ نفر اطرافش اعلام کرده‌اند تغییر می‌دهد. برنامه‌ای بنویسید که شبیه‌سازی را تا $100\ n^2$ تکرار اجرا کند و هر ۱۰۰ تکرار یکبار وضعیت را با رنگ‌آمیزی خانه‌ها با دو رنگ مختلف نمایش دهد. برنامه را به ازای سه مدل متفاوت اجرا کنید. در مدل اول فرض کنید که در شرایط تساوی، یعنی وقتی تعداد رأی اطرافیان ۴ به ۴ است فرد رأی قبلی خود را تغییر نمی‌دهد. در مدل دوم فرض کنید که در شرایط تساوی فرد رأی قبلی خود را تغییر می‌دهد و در مدل سوم فرد رأی خود را در شرایط تساوی تغییر نمی‌دهد ولی در وضعیت ۳ به ۵ به نفع بازنده رأی می‌دهد. n را در حدود ۳۰ تا ۵۰ فرض کنید. برای نمونه به سایت http://ccl.northwestern.edu/netlogo/models/Voting مراجعه کنید.

S3. شبیه‌سازی رفتار فداکاری در گاومیش‌ها.

یک منطقه‌ی چراگاهی را به صورت یک زمین مربعی n×n شامل n^2 مرتع کوچک در نظر بگیرید. هر مرتع محل چرای گاومیش‌هایی است که در آن قرار دارند. هر گاومیش در هر مرحله از شبیه‌سازی مقداری از علف مرتعی که در آن قرار دارد را می‌خورد. این مقدار بر حسب واحد طول (مثلا سانتیمتر)، مقدار علفی است که گاومیش از سطح بالای علفهای مرتع می‌خورد. متناسب با مقدار علف خورده شده به انرژی گاومیش اضافه می‌شود. گاومیش‌ها به آهستگی در اطراف پرسه می‌زنند و در هر مرحله ممکن است تا حداکثر مسافتی دور تر از موقعیت کنونی در یک جهت تصادفی گام بردارند. در هر چرخه از شبیه‌سازی مقدار ثابتی از انرژی هر گاومیش به دلیل سوخت و ساز کاهش می‌یابد. اگر انرژی گاومیش به صفر برسد (یا کمتر از آن شود) می‌میرد و اگر به مقدار آستانه‌ای مشخصی برسد یک گاومیش دیگر از همان نژاد تولید می‌کند و مقدار معینی از انرژی خود را از دست می‌دهد. انرژی اولیه‌ی هر گاومیش جدید ۵۰ است و در یک مکان تصادفی در همان مرتعی که والدش قرار دارد قرار می‌گیرد.

فرض کنید دو نژاد مختلف از گاومیش‌ها داریم. یکی حریص است و بدون توجه به ارتفاع علف آنرا می‌خورد. بدین ترتیب ممکن است علف را حتی تا زمانی که ارتفاعش حداقل به اندازه‌ی یک وعده چرا باشد بخورد. نژاد دیگر فداکار است و علفهایی را که ارتفاعشان از *«آستانه‌ی رشد»* کوتاهتر است نمی‌خورد. اصولا علفهایی که ارتفاعشان از آستانه‌ی رشد بلندتر است در مقایسه با علفهای کوتاهتر از آستانه‌ی رشد با سرعت بیشتری رشد می‌کنند. فرض کنید در هر چرخه از شبیه‌سازی علفهای کوتاهتر از آستانه‌ی رشد به اندازه‌ی معینی (در واحد طول مثل سانتیمتر) بلند می‌شوند و علفهای بلندتر از یا مساوی آستانه‌ی رشد به اندازه‌ی معین (و بزرگتری) رشد می‌کنند. در نهایت رشد علف وقتی به ارتفاع معینی برسد متوقف می‌شود.

برنامه‌ای بنویسید که مدلی با مشخصات فوق را در زمینی به ابعاد ۳۰×۳۰ یا ۵۰×۵۰ شبیه‌سازی کند و تغییرات مدل را در هر ۱۰۰ تکرار یکبار به طور گرافیکی نمایش دهد. برنامه باید امکان تعیین مقادیر مختلف برای پارامترهای مدل را در اختیار کاربر قرار دهد. برای نمونه به سایت http://ccl.northwestern.edu/netlogo/models/Cooperation مراجعه کنید.

S4. تحلیل عددی متن.

برنامه‌ای بنویسید که یک فایل حاوی یک متن به زبان انگلیسی را بخواند و محاسبات زیر را انجام دهد.

- تعداد کلمات غیریکسان در متن را محاسبه کند. دو کلمه را یکسان فرض می‌کنیم اگر بدون توجه به حروف بزرگ و کوچک یکسان باشند. همچنین هر کلمه‌ای که با اضافه شدن پسوند s یا 's یا ing به انتهای کلمه‌ی دیگر بدست آید با آن کلمه یکسان فرض می‌شود. البته لازم نیست از نظر معنایی درست باشد.
- کلماتی را که با حرف بزرگ شروع شده‌اند به جز کلمات اول جمله را به عنوان اسامی خاص در فایل خروجی دیگر لیست کنید و در مقابل هر یک شماره‌ی پاراگراف‌هایی را که کلمه در آن آمده است لیست کنید، مانند نمایه‌ی یک کتاب (فهرست اسامی خاص).
- ۲۰ کلمه‌ای که کمترین تکرار در متن دارند و از اسامی خاص نیستند را به ترتیب تعداد تکرارشان در فایل دیگری لیست کنید. به این لیست ۱۰ کلمه که در بین اسامی خاص دارای کمترین تکرار هستند نیز اضافه کنید.
- هر جفت از کلمات موضوع بند قبل که در متن در فاصله‌ای کمتر از ۵ کلمه با یکدیگر قرار دارند را در فایل خروجی دیگری لیست کنید.

برای اجرای برنامه می‌توانید از نمونه‌هایی که در پیوست ایمیل هستند استفاده کنید.

S5. L-system و گرافیک لاک‌پشتی .

گرافیک لاک‌پشتی (turtle graphics) یک ابزار بسیار مقدماتی برای بیان ترسیمات هندسی در صفحه (و فضا) است. در این شیوه، ترسیمات توسط یک لاک‌پشت فرضی انجام می‌شوند. در هر لحظه، این لاک‌پشت در یک موقعیت صفحه‌ی گرافیکی قرار گرفته است و دارای یک جهت معین برای حرکت به جلو است. وقتی دستور F را به این لاک‌پشت می‌دهید به اندازه‌ی یک گام جلو می‌رود و پاره‌خطی که امتداد حرکتش را نشان می‌دهد ترسیم می‌شود. با دستور + لاک‌پشت بدون آنکه حرکت کند ۹۰ درجه به چپ می‌چرخد. این لاک‌پشت دستورات دیگری را نیز می‌فهمد، که به طور خلاصه از این قرارند؛

F: یک گام حرکت به جلو و ترسیم پاره‌خط حرکت

f: یک گام حرکت به جلو بدون ترسیم پاره‌خط حرکت

B: یک گام حرکت به عقب و ترسیم پاره‌خط حرکت

b: یک گام حرکت به عقب بدون ترسیم پاره‌خط حرکت

+ : چرخش به چپ به اندازه‌ی زاویه‌ی پیش‌فرض

- : چرخش به راست به اندازه‌ی زاویه‌ی پیش‌فرض

<a : چرخش به چپ به اندازه‌ی a درجه

>a : چرخش به راست به اندازه‌ی a درجه

@a : تغییر زاویه چرخش پیش‌فرض به a (بدون حرکت و چرخش). اگر از ابتدا زاویه‌ی پیش‌فرض تعیین نشود مقدار آن ۹۰ درجه است.

] : اطلاعات موقعیت و جهت حرکت را در پشته نگهدار.

[: لاک‌پشت در موقعیت و جهتی قرار می‌گیرد که از پشته بیرون می‌آید.

مثلا رشته زیر یک ستاره‌ی پنج‌پر رسم می‌کند؛

@108>36F+F+F+F+F

در مرحله‌ی اول این پروژه، برنامه‌ای برای پردازش دستورات گرافیک لاک‌پشتی بنویسید. بدین‌ترتیب که کاربر از طریق یک textbox دستورات را وارد می‌کند و ترسیم صورت می‌گیرد. برنامه باید حداقل دو گزینه داشته باشد. گزینه‌ی اول برای پاک کردن و برگرداندن لاک‌پشت به مرکز صفحه و قرار دادن آن در جهت رو به بالا است. گزینه‌ی دیگری انتخاب آن است که اندازه‌ی ترسیمات مطلق باشد یا متناسب با ابعاد صفحه باشد. در حالت اول طول گام حرکت لاک‌پشت توسط کاربر تعیین می‌شود ولی در حالت دوم طول گام باید توسط برنامه به گونه‌ای محاسبه شود که کل شکل ترسیم شده در چارچوب صفحه بگنجد.

زبان L-system گونه‌ای از زبان‌های صوری است که اولین بار توسط یک گیاه‌شناس مجارستانی به نام Lindenmayer برای توصیف الگوی رشد گیاهان ابداع شد. یک L-system شامل تعدادی متغیر جایگزین‌شونده، تعدادی متغیر پایانی و تعدادی قواعد جایگذاری (یا قواعد مولد) است. نکته اصلی در L-system نحوه‌ی تولید کلمات در آن است. بدین ترتیب که با شروع از یک کلمه‌ی اولیه(axiom word)، همه‌ی متغیرهای جایگزین‌شونده در این کلمه بر اساس قواعد جایگذاری داده شده به طور همزمان جایگذاری می‌شوند و کلمه‌ی جدید تولید می‌شود. همین کار بر روی کلمه‌ای که در مرحله‌ی قبلی بدست آمده است تکرار می‌شود. مثلا برای L-system زیر داریم؛

Axiom: @60XF

X -> YF-XF-Y

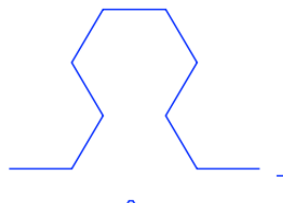
Y -> XF+YF+X

1st iteration: @60YF-XF-YF

2nd iteration: @60XF+YF+XF-YF-XF-YF-XF+YF+XF

3rd iteration: @60YF-XF-YF+ XF+ YF+ XF +YF-XF -Y F- XF +Y F+XF -YF -X F-YF- XF +Y F+ XF -Y F -X F-Y F+X F+ YF +X F+Y F-X F-YF

در اینجا ما فرض می‌کنیم نمادهای گرافیک لاک‌پشتی متغیرهای پایانی و حروف الفبای انگلیسی به جز B, F, b, f متغیرهای جایگزین‌شونده هستند. توجه کنید که نمادهایی به جز نمادهای شرح داده شده در بالا، برای لاک‌پشت بی‌معنا هستند و تغییری در وضعیت آن نمی‌دهند. در زیر شکلی که از کلمه‌ی تولید شده در تکرار دوم بدست می‌آید را می‌بینید.



در مرحله‌ی دوم، این پروژه باید شکلی را که توسط یک زبان L-system تولید می‌شود را ترسیم کند. تعداد تکرارهای جایگذاری برای تولید کلمه توسط کاربر تعیین می‌شود. برای مثال زبان زیر را در نظر بگیرید

Axiom: @25X

X -> F-[[X]+X]+F[+FX]-X

F -> FF

S6. انبارداری.

برای یک پروژه‌ی انبارداری، می‌خواهیم نوع‌داده‌ای مناسبی برای کالاها تعریف کنیم. اطلاعاتی که برای هر کالا مورد توجه هستند عبارتند از: نام کالا، موجودی در انبار و تاریخ ورود به انبار. توجه کنید که ممکن است دو رکورد با نام کالای یکسان ولی تاریخهای ورود به انبار متفاوت وجود داشته باشند. برنامه‌ای بنویسید که اطلاعات ۱۰۰۰ رکورد از کالاها را از ورودی بگیرد و بنابر انتخاب کاربر یکی از گزارش‌های زیر را تولید کند.

لیست مرتب‌شده‌ی کالاها بر حسب تاریخ ورود به انبار. در جاهایی که تاریخ‌ها یکسان است لیست باید برحسب موجودی به طور صعودی مرتب باشد.

لیست مرتب‌شده‌ی کالاها برحسب نام کالا. در جاهایی که نام کالا یکسان است لیست باید برحسب تاریخ ورود به انبار به طور نزولی مرتب‌باشد.

لیست مرتب‌شده‌ی کالاها برحسب تعداد موجود در انبار. در جاهایی که نام کالا یکسان است تنها یک رکورد با مجموع تعداد موجودی آن کالا در انبار و قدیمی‌ترین تاریخ ورود باید در خروجی گزارش شود.

☆ نمره‌ی کامل این پروژه به برنامه‌ای تعلق می‌گیرد که برای هریک از گزارش‌های فوق، الگوریتم مرتب‌سازی بیش از دو بار فراخوانی نشود.

S7. مساحی

فرض کنید مستطیلهایی با اضلاعی موازی محورهای مختصات داریم. برنامه‌ای بنویسید که مختصات گوشه سمت چپ و پائین و گوشه سمت راست و بالای n مستطیل از این نوع را دریافت کرده، مساحت سطح پوشیده شده توسط آنها را محاسبه و چاپ کند. توجه کنید در حالت کلی برخی از مستطیلها ممکن است همپوشانی داشته باشند.

S8. شبیه سازی جامعه مورچگان. Ant colony

زمینی به شکل مستطیل داریم. در این زمین تعدادی مناطق غذایی و تعدادی مانع وجود دارد. شکل مناطق غذایی و سوراخها را با مستطیل یا دایره در نظر بگیرد که با ابعاد و در نقاط متفاوتی بر روی زمین واقع شده اند. رفتار پیچیده ای که مورچه ها در طبیعت برای یافتن غذا انجام می دهند نتیجه ی حاکم بودن تنها چند قانون ساده است. در شبیه سازی ما، این قوانین در عبارات زیر ساده شده اند:

> در هر تکرار از روند شبیه سازی، هر مورچه یک گام به جلو حرکت می کند و مقدار ثابتی فرومون در نقطه ای که ترک کرده است از خود به جای می گذارد.

> سمت حرکت بعدی هر مورچه بطور تصادفی از بین جهت های روبه جلو برای آن تعیین می شود. احتمال انتخاب هر جهت متناسب با میزان فرومونی است که در مقصد گام بعدی وجود دارد. به عبارت ساده مورچه ها عمدتا به سمت بوی فرومون کشیده می شوند. مقصد نباید بر یک مانع واقع باشد. مورچه ها هیچگاه بیش از ۹۰± درجه چرخش نمی کنند.

> میزان فرومون در نقاط صفحه با عبور هر مورچه انباشته می شود ولی بعد از هر l تکرار از شبیه سازی، بخشی از فرومون در هر نقطه تبخیر می شود و نسبت ثابت s از آن باقی می ماند. (s بین ۰ و ۱ و معمولا نزدیک به ۱ است).

> هر مورچه که به منطقه غذا برسد ابتدا یک چرخش ۱۸۰ درجه انجام میدهد سپس حرکت خود را به همان شکل سابق ادامه می دهد.

> هر مورچه اگر t گام پس از یافتن آخرین نقطه غذایی، منبع غذایی جدیدی را پیدا نکند از بین می رود و در عوض مورچه جدیدی را بطور تصادفی در نقطه ای از زمین می گذاریم. توجه کنید که این وضعیت شامل مورچه ای نیز می شود که با یک منبع غذایی دوبار یا بیشتر پشت سر هم در مدت t گام برخورد می کند.

شبیه سازی با n مورچه آغاز می شود و پس از گذشت تعداد مراحل کافی، متناوبا صحنه ای از زمین ترسیم می شود. میزان فرومون انباشته شده در هر نقطه از زمین را می توانید از طریق رنگ آمیزی با شدتهای متفاوت نشان دهید. ترتیبی دهید تا تمام پارامترهای شبیه سازی و همچنین مکان و ابعاد غذاها و موانع توسط کاربر قابل تعیین باشد. حاکم بودن یا نبودن شرط ۵ را نیز برای کاربر اختیاری کنید.

S9. ارزیابی عبارات ریاضی.

ورودی. یک رشته‌ی حرفی شامل یک عبارت ریاضی شامل اعداد صحیح، عملگرهای + - * / و پرانتزها.

خروجی. ارزش عددی عبارت داده شده. مثلا خروجی برنامه برای عبارت $(2*(3+5)-4)*(9-4/2)+1$ مقدار ۸۵ است.