به نام خدا

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| لیست پروژه ها | درس: برنامه نویسی پیشرفته | ترم دوم سال تحصیلی 94-93 | مدرس: مظفر بگ محمدی | دانشگاه ایلام |

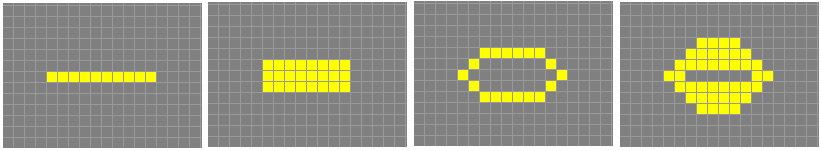
* هدف این پروژه ها آشنایی دانشجو با نحوه ی نوشتن پروژه های کوچک است.
* ظرفیت هر کدام از پروژه ها 2 گروه 2 نفره است و اولویت با کسانی است که زودتر مراجعه کنند.
* تغییر پروژه بعد از اخذ آن امکان پذیر نیست.

**یک: بازی زندگی**

این بازی توسط کامپیوتر انجام می شود. تعدادی سلول بصورت یک ماتریس n\*n وجود دارند. بعضی از سلول ها زنده (رنگ زرد) و بعضی مرده (رنگ خاکستری) هستند. بعد از گذشت یک واحد زمانی (مثلاً یک ثانیه) وضعیت سلولها ها طبق قوانین زیر عوض می شود. بازی با یک هسته ی اولیه شروع می شود و مراحل رشد و نمو و مرگ و میر سلول باید مرحله به مرحله نشان داده شود.

* هر سلول زنده که 1 همسایه ی زنده دارد یا هیچ همسایه ی زنده ای ندارد، از تنهایی می میرد.
* هر سلول زنده ای که 4 یا بیشتر همسایه ی زنده داشته باشد از کمبودهای ناشی از جمعیت زیاد می میرد.
* هر سلول زنده با 2 یا 3 همسایه ی زنده، به حیات خود ادامه می دهد.
* هر سلول مرده که 3 همسایه ی زنده داشته باشد، به دنیا می آید (یا زنده می شود)

مراحل رشد و نمو از چپ به راست نشان داده شده اند.



حداقل 10 حالت جالب که منجر به تولید اشکال زیبا بشود را پیدا کنید و به کاربر اجازه دهید که آنها را بارگذاری کند. باید زمان بین تغییر حالتها قابل کنترل باشد (مثل آهسته، معمولی و سریع). حداقل سایز آرایه را 20 در 40 در نظر بگیرید. اما استفاده از اندازه های خیلی بزرگتر توصیه می شود. تعداد همسایه ها در مرزها کمتر است.

گروه اول: نیکتا حاتمی

**دو: حمله ی اعداد اول:**

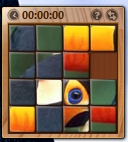
در این بازی یک عدد طبیعی با سرعت ثابت از بالای پنجره به سمت پایین حرکت می کند و کاربر باید عاملهای اول موجود در آنرا با کلیک روی دکمه های مناسب پیدا کند. در قسمت پایین برنامه تعدادی دکمه وجود دارد که شامل اعداد 2، 3، 5، 7، 11، 13 و ... هستند. مثلاً اگر عدد 24 به سمت پایین حرکت کند و کاربر دکمه ی 2 را بزند، عدد به 12 تبدیل می شود. بعد از زدن دکمه ی 3، عدد به 4 تبدیل می شود. کاربر باید دو بار دیگر دکمه ی 2 را بزند تا عدد به 2 و بعد به 1 تبدیل شود و عدد بعدی شروع به حرکت کند.

اعداد به صورت تصادفی انتخاب می شوند و سرعت حرکت آنها به تدریج زیاد می شود. اگر کاربر عامل اول را اشتباه حدش بزند، این عدد به صورت آزاد سقوط کرده و عدد بعدی شروع به حرکت می کند. اگر تعداد اعداد سقوط کرده در یک قسمت از بازی از یک حد مشخص (مثلاً 5) بگذرد، بازی خاتمه پیدا می کند.

در مورد گرافیک بازی می توانید از بازی معروف tetris ایده بگیرید.

**سه: پازل دستی اعداد**

این بازی شامل 16 خانه است که اعداد 1 تا 15 به صورت تصادفی در آنها چیده شده اند و یکی از خانه ها خالی است. کاربر باید با جابجایی خانه خالی اعداد را مرتب کند. برای آشنایی بیشتر می توانید یکی از ویجتهای ویندوز را ببینید. در صورت امکان از یک تصویر به جای اعداد استفاده کنید. برنامه باید قابلیت ثبت بهترین رکوردها (از لحاظ زمانی) را داشته باشد. هم چنین کاربر باید بتواند تا 5 حرکت به عقب برگردد.



**چهار: تطبیق اشکال گرافیکی**

در این بازی، 2n دکمه ی مخفی وجود دارد. زیر هر کدام از دکمه ها یک تصویر وجود دارد و از هر تصویر دقیقاً دو مورد وجود دارد. اگر کاربر روی یک دکمه کلیک کند، تصویر زیر آن مشخص خواهد شد. حال اگر کاربر روی یک دکمه ی دیگر کلیک کند، تصویر زیر آن با تصویر جاری مقایسه می شود، اگر برابر باشند هر دو تصویر رو خواهند شد. در غیر این صورت، تصویر جاری مخفی و نصویر جدید رو خواهد شد. بازی تا وقتی که تمام تصاویر رو شوند ادامه خواهد یافت. برنامه باید هر دفعه تصاویر را بطور تصادفی بچیند. مدت زمانی که کاربر بازی کرده را به او نشان دهد و بهترین زمان را ثبت کند. هم چنین، تعداد حدسهای غلط را بشمارد.

**پنج: مخفی کردن تصویر**

در این پروژه یک فایل bmp و یک فایل متنی وجود دارند. شما باید فاییل bmp را باز کنید و در اطلاعات مربوط به پیکسلها فایل متنی را جایگذاری کنید. بدین ترتیب که هر پیکسل رنگی شامل سه رنگ و 24 بیت یا 3 بایت است. هر بایت یک عدد بین 0 تا 255 است که میزان رنگ مربوطه را مشخص می کند. کدام از بایتها می توانند 1 بیت را در خود حمل کنند. به عنوان مثال، اگر کد رنگ برابر 155 باشد، کم ارزش ترین بیت برابر 1 است. اگر ما بخواهیم بیت 0 را در این رنگ جایگذاری کنیم، باید کم ارزشترین بیت را برابر 0 قرار دهیم. لذا کد رنگ به 154 تغییر می کند. اما اگر بیت مورد نظر ما برای کدگذاری 1 باشد، تغییری در کد این رنگ نمی دهیم.

در حالت کلی، ما فایل متنی را به صورت کاراکتر به کاراکتر می خوانیم. هر کاراکتر شامل 8 بیت است و هر کدام از بیتها در یک کد رنگ قرار می گیرند. لذا، برای کدگذاری سه کاراکتر به هشت پیکسل رنگی نیاز مند هستیم.

**شش: بازی قورباغه:**

در این بازی یک قورباغه باید عرض رودخانه را با پریدن روی برگهایی که با سرعتهای مختلف در حال حرکت هستند را طی کرده و به طرف مقابل برسد. به علاوه تعدادی قایق در جهت مخالف و با سرعتهای متفاوت در حال حرکت هستند که قورباغه نباید روی آنها بپرد یا در آب بیافتد.

گروه اول: مهسا رجبی

**هفت: بازی حافظه:**

در این بازی چهار مربع با رنگهای مختلف برای مدت مشخصی به کاربر نشان داده می‌شود. سپس، رنگ تمام مربعات خاکستری می شود و از کاربر خواسته می شود که رنگ مربعها را به سرعت تعیین کند. مراحل بازی باید به تدریج سختتر شوند. یعنی تعداد رنگها و مربعات زیاد شود.

گروه اول: سینا مرادی و امین ابراهیمی

**هشت: بازی ریاضی:**

در این پروژه باید سوالات ساده ی ریاضی شامل جمع، ضرب، تفریق و تقسیم از کاربر پرسیده شود. هم سرعت و هم دقت در پاسخگویی دارای امتیاز است. سوالات باید دارای پیچیدگیهای مختلف و شامل پرانتز نیز باشند.

**نه: ترکیبات مرتب ارقام:**

می‌خواهیم تمام ترکیبهای ارقام وارد شده را به صورت مرتب چاپ کنیم. مثلا برای ارقام ۰، ۱ و ۲ ترکیبات مرتب سه رقمی به صورت زیر هستند:

012   021   102   120   201   210

در این پروژه k رقم که ممکن است تکراری باشند را از کاربر می گیرید و تمام ترکیبهای n رقمی (n=<k) رقمی آنها را چاپ می‌کنید.

**گروه اول:** سجاد رستمی گروه دوم: میترا حسین پناهی – دلنیا فلاحی

**ده: بزرگترین مجموع:**

در مثلث زیر اگر از راس مثلث شروع کنید و از طریق خانه ‌های مجاور به سمت پایین حرکت کنید. بزرگترین مجموعی که می توانید به دست آورید برابر ۲۳ است.

**3**  
**7** 4  
2 **4** 6  
8 5 **9** 3

حال بزرگترین مجموع را در مثلث زیر پیدا کنید. دقت کنید که ۱۶۳۸۴ حالت مختلف وجود دارد. هدف ما این نیست که تمام حالتهای مختلف را امتحان کنیم، بلکه دنبال یک راه حل هوشمندانه هستیم.

75  
95 64  
17 47 82  
18 35 87 10  
20 04 82 47 65  
19 01 23 75 03 34  
88 02 77 73 07 63 67  
99 65 04 28 06 16 70 92  
41 41 26 56 83 40 80 70 33  
41 48 72 33 47 32 37 16 94 29  
53 71 44 65 25 43 91 52 97 51 14  
70 11 33 28 77 73 17 78 39 68 17 57  
91 71 52 38 17 14 91 43 58 50 27 29 48  
63 66 04 68 89 53 67 30 73 16 69 87 40 31  
04 62 98 27 23 09 70 98 73 93 38 53 60 04 23

**گروه اول**: حسنا درویشی - هانیه صیدکرمی

**یازده:** **حدس کلمه ي عبور:**

فرض کنید رمز عبور شما فقط از ارقام تشکیل شده است. هر بار که می خواهید وارد سیستم شوید کامپیوتر از شما سه رقم را در مکانهای i ام، j ام و k ام می خواهد. مثلاً، اگر رمز عبور 531278 باشد و کامپیوتر از شما رقم دوم، سوم و پنجم را بخواهد، پاسخ صحیح 317 خواهد بود. در فایل <keylog.txt> پنجاه کلید موفقیت آمیز اخیر کاربر ذخیره شده‌اند. با این فرض که i<j<k است، کوچکترین رمز معتبر را که با این فایل همخوانی دارد را پیدا کنید.

**گروه اول:** میلاد مرادی- میلاد بیگ محمدی

**دوازده: بازی سودوکو:**

در فایل <sudoku.txt> پنجاه پازل مختلف قرار دارند که از آسان به سخت مرتب شده اند. اعداد 0 در این فایل نشانگر جاهای خالی هستند که باید برنامه شما آنرا پیدا کند. برنامه‌ای بنویسید که یکی از این پازلها را انتخاب و نمایش دهد و سپس آنرا حل کند.

**گروه اول**: راضیه نرگسی و معصومه شهبازی **گروه دوم**: کیوان مرادخانی

**سیزده: رمزگذاری فایل:**

در این پروژه‌ فرض شده است که می‌خواهیم فایلهایی که حاوی کاراکترهای اسکی هستند را رمز گذاری کنیم. طول کلید رمز برابر سه است و طول فایل نیز دلخواه است. هر کدام از کاراکترهای فایل تبدیل به کد اسکی معادل آن می شود. همین کار را با کاراکترهای کلید رمز نیز انجام می ‌دهیم.سپس، کد اسکی معادل هر کدام از کاراکترهای فایل (مثلاً کاراکتر i ام ) را با کد اسکیِ یکی از کاراکترهای رمز که با آن متناظر است ( یعنی کاراکتر i mod 3 ام) XOR کرده و یک عدد به دست می‌آوریم. حال این عدد را به کاراکتر اسکی معادل آن تبدیل کرده و نتیجه را در فایل رمز شده ذخیره می کنیم. به عنوان مثال، کاراکترهای پانزدهم، شانزدهم و هفدهم فایل به ترتیب با کاراکترهای سوم، اول و دوم فایل متناظر هستند. خاصیت XOR این است که اگر فایل رمز شده را دوباره با کلید رمز رمزگذاری کنیم، فایل اصلی به دست می‌آید. مثلاً، 65 XOR 42 = 107 است و 107 XOR 42 = 65 است. برای فهمیدن نحوه ی کار عملگر XOR (یای انحصاری) می توانید از کتاب مدار منطقی موریس مانو یا اینترنت استفاده کنید. در این پروژه شما فایل داده شده را با کلید رمز سه حرفی داده شده توسط کاربر رمزگذاری می کنید. سپس، فایل رمز شده را دوباره رمزگشایی می‌کنید.

**چهارده: ویرایش فایل**

در این پروژه، تعداد خطوط کد در یک فایل جاوا را می‌شماریم. شما در هنگام شمارش باید از خطوط خالی و توضیحات صرفنظر کنید. دقت کنید که هر خط ممکن است شامل چندین توضیح به صورت /\* … \*/ باشد. توضیحات /\* … \*/  نمی‌توانند تودرتو باشند. ممکن است یک رشته شامل توضیح نیز باشد.

**گروه اول**: فاطمه موسوی تنها **گروه دوم**: نیما رستمی – محسن کریمی مقدم

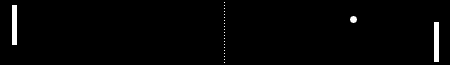
**پانزده: فیلتر بلوم**

در این پروژه فیلتر بلوم را پیاده می‌کنید. فیلتر بلوم به درد پایگاه داده‌های خیلی بزرگ می‌خورد و مشخص می‌کند که آیا داده‌ی مورد نظر در پایگاه داده موجود است یا نه. اما مشکلی که دارد این است که ممکن است فیلتر اعلام کند یک داده در پایگاه داده موجود است در حالی که این گونه نیست. البته بر عکس آن امکان پذیر نیست. یعنی، اگر فیلتر بگوید که داده در پایگاه داده موجود نیست قطعاً درست خواهد بود. نکته‌ی بعدی این که می توانید دقت فیلتر را کنترل کنید.

فیلتر بلوم دارای یک آرایه از بیتها است که در ابتدا همه‌ی آنها برابر صفر هستند. هر بار که یک داده را وارد پایگاه داده کنیم، داده را به یک تابع درهم‌سازی Hash می‌دهیم تا یک عدد تولید شود. حال، تمام بیتهای متناظر با عدد در آرایه را برابر “1” تنظیم می‌کنیم. اگر بخواهیم وجود یک داده در پایگاه داده را چک کنیم بدین صورت عمل می کنیم که Hash داده را محاسبه می‌کنیم. اگر بیتهای متناظر در آرایه تنظیم شده بودند به احتمال خیلی زیاد داده در پایگاه داده موجود است. اگر هنگام ورود داده به پایگاه داده، Hash آن طوری باشد که تمام مکانهای متناظر قبلاً تنظیم شده باشند می گوییم که برخورد رخ داده است. مشکل عدم دقت فیلتر بلوم همین امکان وقوع برخورد است. کلمات وارد شده در پایگاه داده از نوع رشته هستند. لطفاً برای مشخص کردن تابع Hash به اینجانب مراجعه کنید.

**شانزده: بازی پینگ پونگ**

در این پروژه یک بازی پینگ پونگ ساده مثل شکل زیر را طراحی کنید:



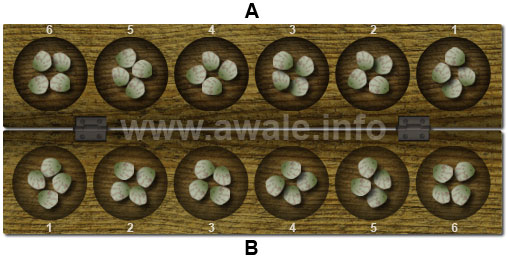
هر کدام از بازیگرها از دو کلید برای حرکت به بالا و پایین استفاده می کنند. به کاربر اجازه دهید که اندازه‌ی راکت و طول و عرض زمین را خودش تنظیم کند.

**هفده: بازی بیست سوالی**

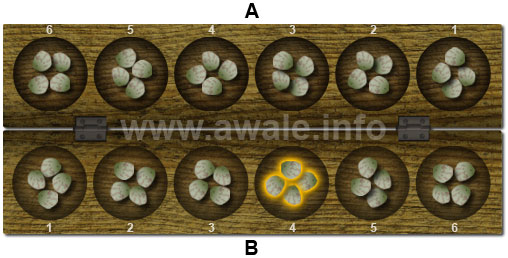
در این پروژه بازی بیست سوالی در مورد حیوانات اجرا می گردد. شما لیست حیوانات را به کاربر نشان می‌دهید و از کاربر می‌خواهید که یکی از حیوانات را در نظر بگیرد. سپس، از کاربر تعدادی سوال می‌پرسید که جواب آنها بله یا خیر است. با توجه به جوابهای کاربر، اسم حیوان را حدس می‌زنید. حداقل تعداد حیوانات ۱۵ است و شما حداقل باید ۳۰۰ سوال مختلف داشته باشید. برای پیاده سازی صحیح این پروژه به مفهوم درخت باینری نیاز خواهید داشت.

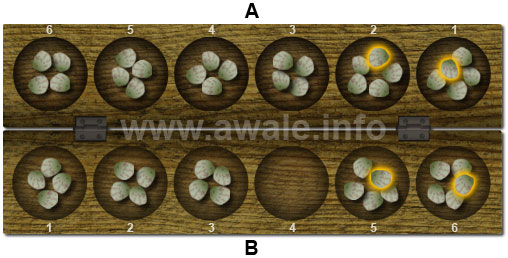
**هجده: بازی آواری**

بازی آواری یکی از بازیهای بومیان آفریقا است که از طریق سنگ و سطل انجام می‌شود. این بازی شامل ۱۲ سطل و ۴۸ سنگ است. سطلهای ۱ تا ۶ مال بازیکن A و سطلهای 7 تا ۱۲ مال بازیکن B است. هدف این بازی این است که سنگهای بیشتری جمع‌آوری شود. اگر بازیکنی بیشتر از ۲۴ سنگ جمع کند می‌تواند پیشنهاد خاتمه ی بازی را بدهد. یا می توان بازی را تا وقتی که نتوان سنگ دیگری را خارج کرد ادامه داد. در ابتدا هر کدام از سطلها دارای ۴ سنگ است.



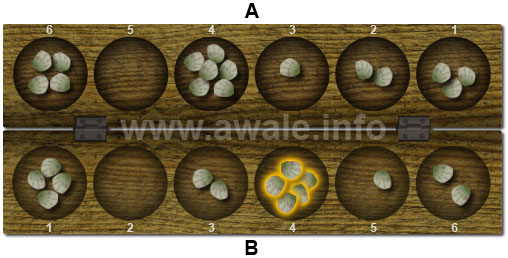
قانون اول: هر بار که نوبت به یک بازیگر رسید، آن بازیگر یکی از خانه های خود را انتخاب می کند و سنگهای آنرا بر می‌دارد و آنها در جهت خلاف حرکت عقربه های ساعت در سطلها بخش می کند. به هر سطل فقط یک سنگ اضافه می‌شود و در صورتی که سطلهای خودمان تمام شود از سطلهای حریف استفاده می کنیم. به شکل زیر دقت کنید:

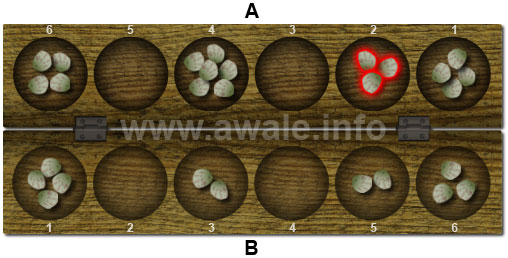
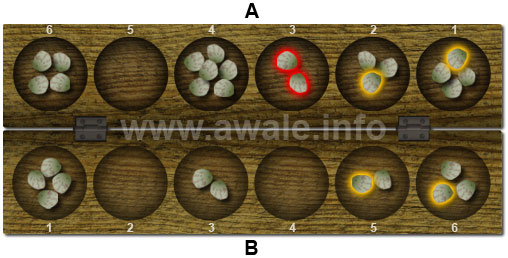




در اینجا، بازیگر B سطل ۴ را انتخاب و سنگهای آنرا پخش کرده است.

قانون دوم: پس از این که بازیگر سنگهای خود را پخش نمود، اگر آخرین خانه ای که به آن سنگ اضافه کرده است متعلق به حریف بود و تعداد سنگهای ۲ یا ۳ بود، آن سنگها را برای خود بر می‌دارد. سپس، به خانه ی قبلی نگاه می‌کند. اگر تعداد سنگهای این سطل هم ۲ یا سه بود آنها را بر می‌دارد. اینکار را آنقدر تکرار می کنیم که به خانه ای برسیم که تعداد سنگهای ۲ یا ۳ نباشد. به شکل زیر دقت کنید:





قانون سوم: اگر سطلی که سنگهای آنرا برداشته‌اید دوازده سنگ یا بیشتر داشته باشد، در اینصورت می‌توان یک دور در تمام خانه ها سنگ قرار داد. اما حق ندارید سنگ دوازدهم را در سطل خالی شده بگذارید و از آن سطل صرفنظر کنید.

قانون چهارم: حق ندارید در یک حرکت تمام سطلهای حریف را خالی کنید. انجام اینکار به منزله ی قبول شکست از طرف شما است.

قانون پنجم: اگر ادامه ی بازی عاقلانه نباشد یا امکان برداشتن سنگ دیگری نباشد، دو طرف می‌توانند بازی را خاتمه دهند و کسی که سنگ بیشتری جمع کرده است برنده است.