برنامه‌سازی پیشرفته

تمرین شماره ۲

مدرس: رامتین خسروی

طراح: محمد مهدوی‌دوست

موعد تحویل: ۱۱ اسفند ۱۳۹۳

سودوکو

هدف از این تمرین آشنایی با تکنیک طراحی بالا به پایین[1] و استفاده از آن در کنار کتابخانه‌های گرافیکی است.

سودوکو یک جدول ۹×۹ از اعداد است که امروزه یکی از سرگرمی‌های رایج در کشورهای مختلف جهان به شمار می‌آید. جدول اصلی این بازی خود به ۹ ناحیه کوچکتر (۳×۳) تقسیم می‌شود و مجموعه‌ای از اعداد به صورت پیش‌فرض در آن قرار می‌گیرند. در حل جدول‌های سودوکو یک اصل باید رعایت شود و آن اینکه اعداد ۱ تا ۹ در هر **سطر**، **ستون** و **ناحیه ۳×۳** جدول بدون تکرار قرار گیرند. برای کسب اطلاعات بیشتر می‌توانید به صفحه توضیحات این بازی در ویکیپدیا مراجعه کنید.[2]
در این تمرین قصد داریم در سه مرحله، نسخه گرافیکی این بازی را طراحی و پیاده‌سازی کنیم.

قسمت اول

برنامه‌ای بنویسید که با دریافت اطلاعات یک جدول سودوکو، مشخص کند که این جدول یک پاسخ صحیح سودوکو است یا خیر. توجه کنید که صحیح بودن جدول وابسته به رعایت اصلی‌ست که در بخش قبل شرح دادیم.
اولین خط ورودی عدد مثبت n است که تعداد جدول‌ها را مشخص می‌کند. اطلاعات هر جدول در ۹ خط ورودی استاندارد قرار می‌گیرند که به ترتیب متناظر با سطرهای اول تا نهم جدول اصلی هستند. در هر خط نیز ۹ کاراکتر داریم که مقادیر ستون‌های جدول در آن سطر خاص را مشخص می‌کنند. توجه کنید که این کاراکترها با فاصله[3] از هم جدا شده‌اند و برای مشخص کردن خانه‌های خالی جدول از # استفاده می‌کنیم. به ازای هر جدول در ورودی، یک خط در خروجی چاپ می‌شود. بسته به صحیح بودن جدول عبارت true یا false نمایش داده می‌شود. نمونه‌ای از ورودی و خروجی این برنامه به شکل زیر است:

خروجی نمونه	ورودی نمونه
false	1 5 3 4 6 7 8 9 1 2 6 7 2 1 9 5 3 4 8 1 9 8 # 4 2 5 6 7 8 5 9 7 6 1 4 2 3 4 2 6 8 5 3 7 9 1 7 1 3 9 2 4 8 5 6 9 6 1 5 3 7 2 8 4 2 8 7 4 1 9 6 3 5 3 4 5 2 8 6 1 7 9

[1] Top Down Design
[2] http://en.wikipedia.org/wiki/Sudoku
[3] Space

قسمت دوم

هدف این قسمت، نوشتن این بازی با واسط کاربری کنسول است طوری‌که از توابع برای شکست بالا به پایین برنامه استفاده شود. در این قسمت ورودی و خروجی به شکل متنی است و در قسمت بعدی گرافیک اضافه خواهد شد. الگوریتم کلی این بازی به این شکل است:

خانه‌های جدول را با مقادیر اولیه خوانده شده از ورودی استاندارد مقداردهی کن.
تا زمانی که جدول حل نشده است تکرار کن.
عدد مورد نظر کاربر و محل قرارگیری آن را از ورودی استاندارد بخوان.
اگر درخواست معتبر است پس از به روز رسانی جدول آن را به کاربر نمایش بده،
در غیر اینصورت پیام خطا مناسب را نمایش بده.

این بازی را با رعایت نکات زیر پیاده‌سازی کنید:

- تا حد امکان عملیات را در قالب توابع مستقل انجام دهید. مثلاً بند اول (مقداردهی اولیه خانه‌های جدول) را در یک تابع مستقل پیاده‌سازی کنید. توابع شما ممکن است خودشان از توابع کوچکتری استفاده کنند.
- به هیچ‌وجه از متغیرهای سراسری استفاده نکنید. هر تابع اطلاعات مورد نیاز را از پارامترهایش دریافت و خروجی‌های خود را از طریق مقدار بازگشتی یا پارامترهای خود به بیرون برگرداند.
- در نامگذاری خوانا برای توابع و پارامترها دقت کنید.

ورودی و خروجی نمونه

۹ خط ابتدایی ورودی استاندارد مقداردهی اولیه جدول را مشخص می‌کنند. فرمت این خطوط مشابه قسمت اول است. درخواست‌های بعدی کاربر با فرمت x y n در هر سطر وارد می‌شوند. x و y محل قرارگیری عدد (به ترتیب ستون و سطر) را تعیین می‌کند. با فرض این‌که خانه بالا سمت چپ خانه‌ی ۱ و ۱ در نظر گرفته می‌شود. n نیز عدد مورد نظر کاربر را مشخص می‌کند. توجه کنید که برنامه شما در صورت وجود خطا نباید متوقف شود و باید پیام متناسب با خطا را مطابق با جدول شماره ۱ به کاربر نمایش دهد. در نظر داشته باشید که منظور از خطا، اشتباه‌های منطقی کاربر نیست. به طور مثال اگر کاربر در ستونی که ۹ وجود دارد مجددا عدد ۹ را قرار داد، نیازی نیست این مساله به او تذکر داده شود و یا عدد درخواستی در جدول قرار داده نشود (مانند ورودی دوم در بخش زیر). تنها پس از تکمیل شدن جدول لازم است صحت یا عدم صحت چیدمان اعداد را به ترتیب با چاپ پیام‌های valid یا invalid به کاربر اعلام نمایید و اجرا برنامه را متوقف کنید. در صورتی که ورودی کاربر منجر به بروز خطا یا اتمام بازی نشد جدول به‌روزرسانی شده را در خروجی استاندارد چاپ نمایید. همچنین برای خوانایی محتوای چاپ شده، پس از پایان رسیدگی به هر درخواست کاربر، یک خط خالی در خروجی استاندارد چاپ نمایید.

تعداد یا فرمت نادرست پارامترها در ورودی	ERR1
x و یا y خارج از صفحه باشند	ERR2
n بین ۱ و ۹ نباشد	ERR3

جدول شماره ۱

نمونه‌ای از ورودی و خروجی این برنامه به شکل زیر است:

ورودی نمونه	خروجی نمونه
5 3 # # 7 # # # # 6 # # 1 9 5 # # # # 9 8 # # # # 6 # 8 # # # 6 # # # 3 4 # # 8 # 3 # # 1 7 # # # 2 # # # 6 # 6 # # # # 2 8 # # # # 4 1 9 # # 5 # # # # 8 # # 7 9 3 1 12 6 1 9 6 1 8 3 1 A 3 21 9	ERR1 5 3 - - 7 9 - - - 6 - - 1 9 5 - - - - 9 8 - - - - 6 - 8 - - - 6 - - - 3 4 - - 8 - 3 - - 1 7 - - - 2 - - - 6 - 6 - - - - - 2 8 - - - - 4 1 9 - - 5 - - - - - 8 - - 7 9 5 3 - - 7 8 - - - 6 - - 1 9 5 - - - - 9 8 - - - - 6 - 8 - - - 6 - - - 3 4 - - 8 - 3 - - 1 7 - - - 2 - - - 6 - 6 - - - - - 2 8 - - - - 4 1 9 - - 5 - - - - - 8 - - 7 9 ERR1 ERR2

قسمت سوم

در این قسمت از کتابخانه‌ی RSDL برای پیاده‌سازی یک واسط کاربر گرافیکی استفاده کنید. کاربر باید بتواند با فشردن کلیدهای جهت[4] بین خانه‌های جدول حرکت کند و قرار گرفتن روی یک خانه باید از نظر دیداری آشکار باشد. به طور مثال می‌توانید رنگ پس‌زمینه خانه را عوض کنید. زمانی که کاربر روی یک خانه قرار دارد می‌تواند با فشردن یکی از کلیدهای ۱ تا ۹ روی صفحه کلید، خانه مورد نظر را مقداردهی کند. توجه کنید که مشابه قسمت دوم، نیازی نیست اشتباه‌های منطقی کاربر را به او تذکر دهید. تنها در پایان بازی، باید درست/غلط بودن جدول را نشان دهید.

[4] Arrow keys

قسمت امتیازی

هنگام شروع بازی، کاربر بتواند قابلیت کمک‌های بصری[5] را فعال کند. در صورت فعال بودن این قابلیت، هنگامی که کاربر یک اشتباه منطقی انجام دهد (به طور مثال در سطری که ۸ وجود داشته‌است، مجددا عدد ۸ را قرار دهد) این مساله به او تذکر داده شود. به طور مثال، رنگ پس‌زمینه دو خانه قرمز می‌شود. برای فعال‌سازی این قابلیت هنگام اجرا از دستوری مشابه عبارت زیر استفاده می‌شود:

./a.out -aid

در نظر داشته باشید که امتیاز این بخش تنها زمانی به شما تعلق می‌گیرد که هر سه قسمت قبلی را به صورت کامل پیاده‌سازی کرده باشید.

نحوه‌ی تحویل

فایل‌های متناظر با هر قسمت را به ترتیب با نام‌های A2-SID-1.cpp، A2-SID-2.cpp، A2-SID-3.cpp و A2-SID-bonus.cpp ذخیره کنید. سپس آن‌ها را در یک پوشه به نام A2-SID قرار داده و آن را با فرمت zip آرشیو کنید و در نهایت فایلی با نام A2-SID.zip را در سایت درس آپلود کنید. (SID پنج رقم آخر شماره‌ی دانشجویی شماست. به عنوان مثال اگر شماره‌ی دانشجویی شما ۸۱۰۱۹۳۱۲۳ است، نام فایل شما باید A2-93123.zip باشد.)

تحویل این تمرین به صورت حضوری است و در هنگام تحویل باید به تمام قسمت‌های کد خود مسلط باشید.

بارم‌بندی

برای این تمرین ۲۰ نمره امتیازی در نظر گرفته شده است.

ساختار و عملکرد مناسب بخش اول	ساختار و عملکرد مناسب بخش دوم	ساختار و عملکرد مناسب بخش سوم	بخش امتیازی	جمع
۱۵ نمره	۳۵ نمره	۵۰ نمره	۲۰ نمره	۱۲۰ نمره

[5] Visual Aid

دقت کنید

- برنامه‌ی شما باید در سیستم‌عامل لینوکس نوشته و با مترجم g++ ترجمه شود.
- در چاپ کردن خروجی نهایت دقت را به خرج دهید.
- به فرمت و نام فایل‌های خود دقت کنید. در صورتی که هر یک از موارد گفته شده رعایت نشود، نمره‌ی صفر برای شما در نظر گرفته می‌شود.
- در صورت کشف تقلب در کل و یا قسمتی از تمرین، برای هر دو طرف نمره‌ی ۱۰۰ ـ منظور خواهد شد.