به نام خدا

دانشکده‌ی مهندسی برق و کامپیوتر دانشکده فنی دانشگاه تهران

مبانی کامپیوتر و برنامه‌نویسی

نیمسال اول ۰۳-۰۲	پروژه‌ی ۲ (UT-Fall)	استاد : دکتر مرادی، دکتر هاشمی

این پروژه دارای بخش امتیازی می‌باشد. همچنین این پروژه دارای بخش تحویل الگوریتم نیز می‌باشد. هدف این پروژه، استفاده از تابع، آرایه و Header Files و نحوه کار با آن ها می‌باشد.

لطفا حتما مورد سوم و چهارم در بخش تحویل را مطالعه نمایید.

اساتید دانشکده برق و کامپیوتر پس از کلاس های طولانی، به استراحت و سرگرمی در دفاتر خود نیاز دارند! پس از برگزاری جلسات متعدد، به این نتیجه رسیدند که از دانشجویان درس مبانی بخواهند تا یک بازی برایشان طراحی کنند تا در اوقات فراغت، با آن سرگرم شوند. در ادامه، شرح بازی، قوانین و نحوه پیاده سازی آن توضیح داده شده است.

شرح بازی

در این بازی قرار است تا شکل های مختلف به صورت بلوکی، از بالا صفحه بازی به پایین در حال حرکت باشند. با استفاده از کلیدهای A و D، موقعیت این شکل ها تنظیم می‌گردد. (D: سمت راست و A: سمت چپ) همچنین با استفاده از کلید space، شکل ها می‌توانند به صورت ۹۰ درجه در جهت عقربه های ساعت بچرخند.

این بازی، یک بازی امتیازی است و امتیاز آن به این صورت است که هنگامی که یک ردیف از بلوک ها در کنار یکدیگر قرار گرفتند، کاربر ۱۰ امتیاز دریافت می‌کند. همچنین پس از قرار گرفتن یک ردیف از بلوک ها، آن ردیف حذف شده و تمام بلوک های دیگر یک ردیف به پایین می‌آیند. (اگر چند ردیف همزمان کامل شد، باید به تعداد ردیف های کامل شده سایر بلوک ها به پایین بیایند.)

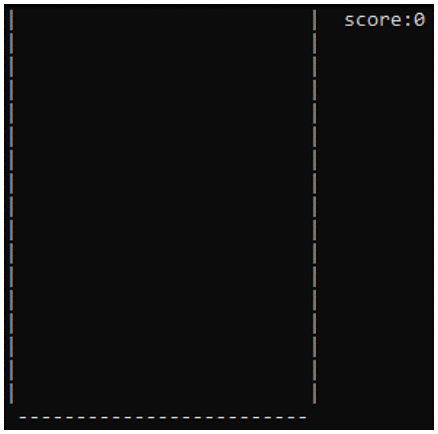
این بازی تا جایی ادامه پیدا می‌کند که بازیکن نتواند اشکال را به ترتیب درستی، روی یکدیگر بچیند و ردیف های کاملی به دست نیاید. در این صورت بازیکن بازنده اعلام می‌شود و امتیاز او در انتها نمایش داده می‌شود.

قوانین بازی

- جابجایی موقعیت اشکال در صفحه بازی حتما باید با کلیدهای D و A صورت گیرد.
- چرخش اشکال حتما باید با کلید space انجام شود.
- اشکال در حال سقوط، باید به صورت تصادفی تولید گردند. (نباید ترتیب خاصی برای اشکال بعدی وجود داشته باشد.)
- پس از کامل شدن یک ردیف از صفحه با بلوک های اشکال، این ردیف باید از بین رفته و تمامی بلوک های درون صفحه بازی یک ردیف به پایین بیایند.
- در صورتی که بلوک های روی هم گذاشته شده، به بالای صفحه بازی برسد، بازیکن بازنده اعلام می‌شود.

نحوه پیاده سازی

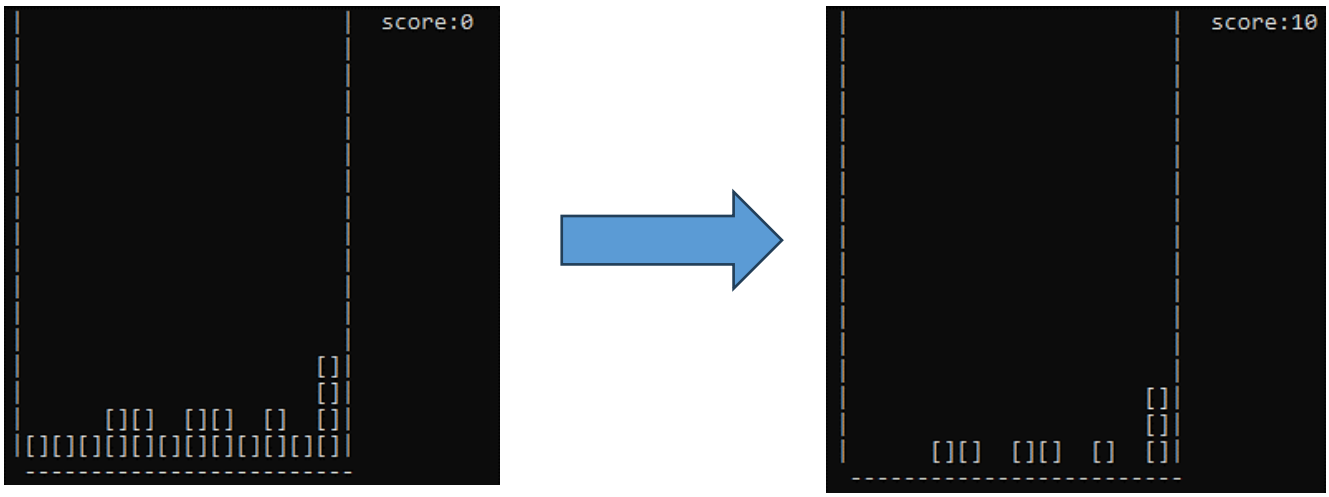
در ابتدا نیاز است تا صفحه بازی طراحی شود تا بازیکن بتواند در آن، اشکال بلوکی را روی یکدیگر قرار دهد. صفحه بازی باید به شکل زیر باشد:



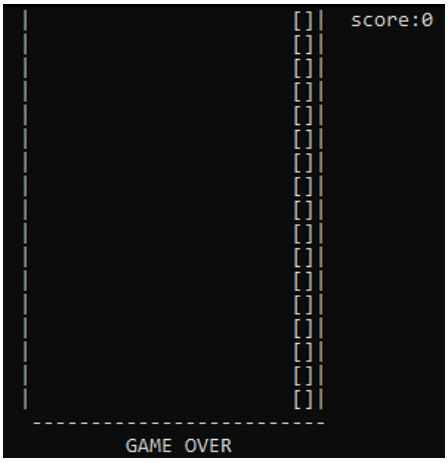
سعی شود تا صفحه بازی در گوشه صفحه cmd قرار نگیرد و کمی وسط صفحه قرار گیرد. شکل هایی که در بازی موجود است، به ۴ شکل مختلف به شکل های زیر تقسیم می‌شوند:

تمامی اشکال باید از بالای صفحه بازی و در وسط آن، شروع به سقوط کنند. در یک زمان، تنها یک شکل باید در صفحه بازی وجود داشته باشد. **هیچگاه دو شکل نمی‌توانند بطور همزمان در صفحه وجود داشته باشند.**

در صورتی که یک ردیف از زمین بازی، بطور کامل با بلوک های اشکال مختلف پر شد، آن ردیف باید حذف شده و تمامی ردیف های دیگر یک واحد به پایین بیایند و به امتیاز بازیکن اضافه شود. برای مثال:



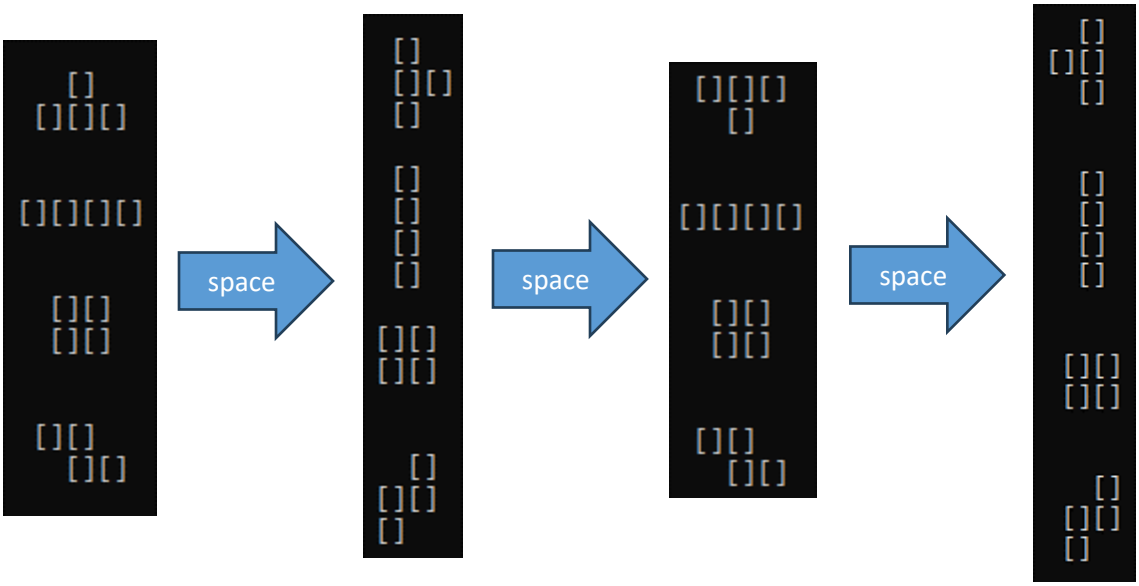
همچنین در صورتی که صفحه بازی پر شود یا تنها یک ستون از آن به بالای صفحه برسد، بازیکن بازنده اعلام می‌شود و بازی به اتمام می‌رسد. (توجه شود که برنامه نباید تمام شود و با فشردن کلیدی یا بستن صفحه cmd برنامه به اتمام می‌رسد.)



- شما می‌توانید ابعاد زمین را تغییر دهید. اما دقت کنید این ابعاد باید منطقی و معقول باشد بطوریکه طول و عرض صفحه بازی باید به صورتی باشد که شرط باخت بتواند به راحتی بررسی شود. برای راحتی می‌توانید عرض صفحه بازی (فاصله دو خط عمودی راست و چپ) را برابر با ۲۰ تا space در نظر بگیرید. برای طول صفحه بازی نیز در همین حدود می‌توان خط عمودی قرار داد.
- می‌توانید قبل از شروع بازی و یا در بالای صفحه بازی، یک پیام خوشامدگویی نیز نمایش دهید. نمایش پیام ها و موقعیت آن ها در این پروژه، برعهده خودتان می‌باشد.
- در صورت برد یا باخت، برنامه باید پیغام مناسبی را نمایش دهد و برنامه نباید به اتمام برسد. این پیام نمایش داده شده می‌تواند ابتکاری باشد.
- پیام ها باید به یکی از دو زبان انگلیسی یا فینگلیش نوشته شود.
- سرعت سقوط اشکال باید به نحوی باشد که بتوان برای قرار گیری آن در صفحه به راحتی تصمیم گرفت و به نحوی باشد که برای رسیدن به پایین صفحه زمان زیادی نبرد.
- برای ورودی گرفتن، می‌توانید از تابع getch که در کتابخانه conio.h قرار دارد استفاده کنید.

برای نمایش صفحه بازی، بهتر است از یک آرایه دوبعدی استفاده نمایید. بطوریکه در این آرایه، مشخص می‌شود که در مختصات i, j صفحه، باید بلوکی از شکل چاپ گردد یا خالی از شکل باشد. در نهایت نحوه ذخیره سازی و پیاده سازی **برعهده خودتان** است.

برای چرخش اشکال در جهت عقربه های ساعت، هر یک از اشکال پس از فشردن کلید space، باید به صورت زیر تغییر شکل پیدا کنند (محدودیتی در تعداد چرخش وجود **ندارد**):



در **صورت نیاز** می‌توانید از قطعه کدهای زیر که به عنوان نمونه می‌باشند، برای انجام بخشی از پروژه خود استفاده نمایید.

```c
int Random(){
    /*
        This function generates a random number between 0 and 3.
        It can be used to determine which shape should fall.
    */
    srand(clock() + time(NULL) + M_PI);
    return rand() % 4;
}
```

```c
void Input(){
    // This function identifies and handles both valid and invalid inputs.
    char input;
    while(input = getchar()){
        if(input == 'd'){
            printf("the pressed key is d\n");
        }
        else if(input == 'a'){
            printf("the pressed key is a\n");
        }
        else{
            printf("the key is undefined\n");
        }
        fflush(stdin);
    }
}
```

```
void rotate(int shape[9], int playground[20][25]){
    /*
    This code handles the rotation of a straight line shape, from [] to [][][][].
                                                                []
                                                                []
                                                                []
    The algorithm first checks for collisions or boundary breaches. It looks at multiple positions: playground[shape[2]][shape[3] + 1],
    playground[shape[2]][shape[3] - 1], playground[shape[2]][shape[3] - 2], and boundary conditions shape[3] - 2 < 0 and shape[3] + 1 > size_y.
    If any of these checks fail (i.e., they return true), the rotation cannot be executed, and the function returns.
    Assuming there is no collision, the code proceeds to clear the current position of the shape by setting
    playground[shape[0]][shape[1]], playground[shape[4]][shape[5]], and playground[shape[6]][shape[7]] to 0.
    The shape's state is then updated for its new orientation: shape[8] is set to 1, indicating a change in the shape's orientation.
    The x-coordinates of the blocks (shape[0], shape[4], shape[6]) are all set to shape[2], aligning them in the same column.
    The y-coordinates are adjusted (shape[1], shape[3], shape[5], shape[7]) to spread the blocks in a horizontal line,
    with the original shape[3] value acting as a pivot point.
    */
    if (playground[shape[2]][shape[3] + 1] || playground[shape[2]][shape[3] - 1] || playground[shape[2]][shape[3] - 2] ||
        shape[3] - 2 < 0 || shape[3] + 1 > size_y)
    {
        return;
    }
    else
    {
        playground[shape[0]][shape[1]] = playground[shape[4]][shape[5]] = playground[shape[6]][shape[7]] = 0;
        shape[8] = 1;
        shape[0] = shape[4] = shape[6] = shape[2];
        shape[1] = shape[3] - 2;
        shape[3] = shape[3] - 1;
        shape[5] = shape[3] + 1;
        shape[7] = shape[3] + 2;
    }
}
```

در قطعه کد بالا، آرایه playground یک آرایه دوبعدی برای نمایش صفحه بازی می‌باشد و ارایه shape که یک آرایه ۹ تایی است، یک آرایه برای مختصات شکل ها در صفحه بازی می‌باشند. (محور x)

محتوای آرایه shape به این صورت است: ایندکس ۰ تا ۷: مختصات هر براکت برای هر بلوک شکل، ایندکس ۸: یک عدد است که نشان می‌دهد شکل فعلی کدام شکل از ۴ شکل است و در چه موقعیتی از چرخش قرار دارد.

توجه داشته باشید که برای استفاده از قطعه کدهای بالا، نیاز است تا در ابتدای فایل خود، کد زیر را بنویسید.

```
#define _USE_MATH_DEFINES

#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#include<math.h>
```

امتیازی

ممکن است بعضی از اساتید در این بازی بسیار ماهر باشند و بخواهند رکوردهای قبلی خود را بشکنند. برای این کار نیازمند دو ویژگی اضافی در این بازی خواهند بود.

بخش اول (تغییر سرعت)

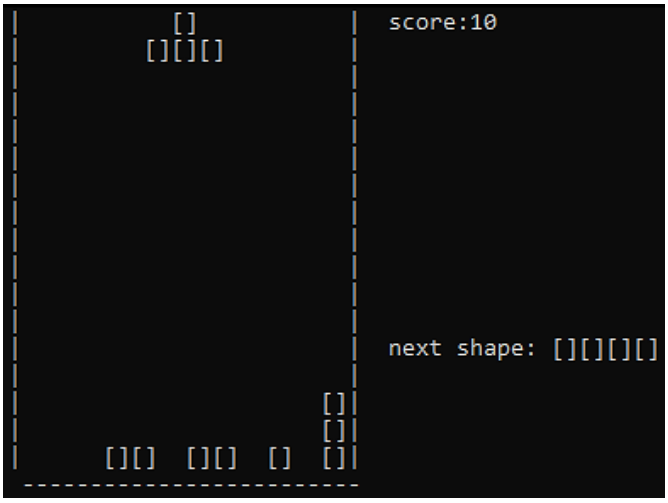
در این بخش بازیکن باید بتواند با فشردن S، سرعت سقوط شکل را بیشتر کند. همچنین بازیکن باید قبل از شروع بازی مشخص کند که با هر بار زدن کلید space، شکل چند درجه در جهت عقربه های ساعت می‌چرخد. (برای مثال می‌توان گفت با وارد کردن عدد ۰ شکل ۹۰ درجه چرخش خواهد داشت یا با عدد ۱ شکل ۱۸۰ درجه چرخش خواهد داشت و ...)

- با هربار فشردن S، سرعت سقوط شکل تا زمانی که شکل در محل خود قرار گیرد افزایش میابد و برای شکل بعد، سرعت باید به حالت قبل برگردد.

- هر چه کلید S بیشتر فشرده شود، سرعت نیز بیشتر می‌شود اما باید توجه داشت که باید یک حد بالا برای سرعت در نظر گرفت و نمی‌توان به صورت نامحدود آن را افزایش داد.
- سرعت افزایش یافته نباید به حدی باشد که ناگهان شکل به انتهای صفحه برسد و باید بتوان مسیر حرکت شکل را مشاهده کرد.

بخش دوم (شکل بعدی)

در این بخش بازیکن باید بداند که شکل بعدی، کدام یک از ۴ حالت ممکن است. برای این بخش می‌توانید یک قسمت در کنار صفحه بازی طراحی کنید و در آن بخش، شکل بعدی ای را که قرار است سقوط کند را نمایش دهید. برای مثال:



- توجه شود که شکل بعدی، باید بلافاصله بعد از سقوط شکل فعلی نمایش داده شود.

چند نکته...

۱. استفاده از هرگونه توابع C++ **ممنوع** می‌باشد و نمره ای به آن تعلق نمی‌گیرد.
۲. پروژه باید نسبت به ورودی های نامعتبر **مقاوم** بوده و در صورت وارد کردن ورودی نامعتبر، عبارت مناسبی را نمایش دهد. توجه شود که در صورت وارد کردن ورودی نامعتبر، برنامه نباید به اتمام برسد و پس از نمایش عبارت مناسب، باید دوباره ورودی بگیرد و بازی پیش برود.
۳. بهتر است هرکدام از مراحل مختلف بازی را در تابع های جداگانه بنویسید و از آن‌ها استفاده کنید. برای مثال می‌توانید یک تابع برای نمایش صفحه‌ی بازی بنویسید و در مرحله‌ی نمایش صفحه، تنها آن را صدا بزنید.
۴. بهتر است که از یک حلقه کنترلی کلی برای تکرار مراحل کلی بازی استفاده کنید.
۵. توجه داشته باشید که شما باید در این پروژه، توابع خود را در فایل های جداگانه با هدرفایل های مخصوص خود بنویسید. شیوه‌ی تقسیم بندی توابع به خودتان مربوط است و هر روش قابل قبولی، پذیرفتنی است.
۶. این توابع ممکن است در طول پروژه به شما کمک کنند: kbhit، sleep، getch

شیوه‌ی نمره دهی

عنوان	نمره
نامگذاری مناسب و اصولی متغیرها	۵
استفاده از تمام ورودی های تابع در آن و ناگذاری مناسب توابع	۵
عدم وجود قطعه کد تکراری	۵
نمایش درست صفحه بازی در command line	۵
دریافت صحیح ورودی در command line	۵
رسیدگی به خطاها	۱۰
نمایش پیام های مناسب	۵
پیاده سازی الگوریتم درست برای تغییر موقعیت اشکال و چرخش آن ها	۱۰
پیاده سازی الگوریتم درست برای تشخیص باخت و حذف ردیف کامل شده	۵
رندوم بودن شکل های تولیدی	۵
کامنت گذاری مناسب در هر جایی که نیاز به مستندسازی دارد	۵
استفاده از header file	۵
فلوچارت الگوریتم روند پروژه	۱۰
تست و اجرای برنامه	۲۰
امتیازی بخش اول	۱۰
امتیازی بخش دوم	۲۰
مجموع	۱۳۰

تحویل

۱. تنها فایل‌های با فرمت ".c" و ".h" را در یک فایل زیپ با فرمت ".zip" و با نام CA۲-SID.zip قرار دهید که SID همان شماره دانشجویی شماست. برای مثال اگر شماره دانشجویی شما ۸۱۰۱۰۲۱۲۳ باشد، باید نام فایل خود را CA۲-۸۱۰۱۰۲۱۲۳.zip قرار دهید و آن را در قسمت درنظر گرفته شده در صفحه درس در سامانه ایلرن آپلود نمایید.

۲. برنامه های شما باید با زبان برنامه نویسی C نوشته شود و استفاده از دیگر زبان های برنامه نویسی مجاز **نیست.**

۳. برای این پروژه و پروژه های بعدی، به جای تمدید پروژه، برای شما ۸ روز گریس پس از ددلاین (تحویل بدون تاخیر و بدون کسر نمره) در نظر گرفته شده است. می‌توانید از این ۸ روز برای تحویل این پروژه ها استفاده کنید و متناسب با شرایط خود، از آن استفاده کنید. توجه داشته باشید ماکزیمم استفاده از گریس برای هر پروژه، ۵ روز می‌باشد و پس از آن تحویل پروژه به هیچ وجه امکان پذیر **نیست**. لطفا در مدیریت گریس های خود توجه کنید.

۴. کسر گریس به صورت ساعتی خواهد بود بدین معنا که شما در مجموع ۱۹۲ ساعت گریس در اختیار دارید و به ازای هر ساعت تاخیر در آپلود پروژه، از آن کسر خواهد شد. تمام تاخیرها به بالا گرد می‌شود بدین معنی که اگر نیم ساعت تاخیر داشتید، یک ساعت از گریس شما کسر خواهد شد.

۵. لازم است تا برای این پروژه، فلوچارت الگوریتم روند پروژه نیز طراحی شده و آپلود گردد. مهلت آپلود فلوچارت تا **ساعت ۲۳:۵۵ یکشنبه ۱۹ آذر** می‌باشد.

۶. همچنین مهلت آپلود بخش کد پروژه، تا **ساعت ۲۳:۵۵ سه شنبه ۲۸ آذر** است.

۷. **این پروژه دارای تحویل حضوری می‌باشد**. لازم است تا در زمان تحویل، بر پروژه و کد خود تسلط کامل داشته باشید. پروژه ها برای یادگیری برنامه‌نویسی و مباحث مطرح شده در کلاس طراحی می‌شوند و انجام آنها به صورت انفرادی خواهد بود. همچنین، در صورت شباهت میان دو پروژه (که به وسیله ی نرم افزارهای مربوطه چک می‌شود) برای هر دو نفر نمره‌ی صفر در نظر گرفته خواهد شد.

۸. در صورت وجود هرگونه سوال می‌توانید پرسش های خود را در فروم درس (در بخش مربوط به این پروژه) مطرح نمایید و یا از طریق ایمیل یا تلگرام با سهیل در ارتباط باشید.

موفق و سربلند باشید