توضیحات کد پایه (agent) نخستین دوره مسابقات مسابقات هوش مصنوعی دانشگاه شاهد – نقطه و خط

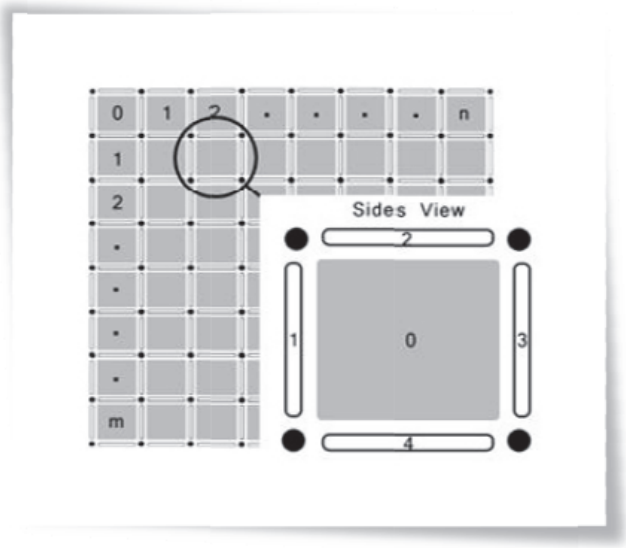
توضیحات ابتدایی

برای کامپایل کردن این کد می بایست از کامپایلر استاندارد مانند g++ یا visual studio (ویرایش ۲۰۰۵ به بعد) استفاده نمایید.

در قسمت هایی از متن زیر، واژه سلول بجای اضلاع و خانه های صفحه بازی استفاده شده است. از واژه سلول بیشتر در هنگام توضیح کدها استفاده شده است (به دلیل وجود کلاس Cell در کد عامل).

کلاس Cell

تعریف این کلاس به شکل زیر است:




```
class Cell
{
public:
      int Row;
      int Column;
      int Side;
};
```

این کلاس برای نگه داری اطلاعات یک سلول از زمین بازی است. همان طور که در شکل مقابل نیز مشخص است هر سلول متشکل از یک خانه و ۴ ضلع است. در حقیقت باید برای سلول ها تصمیم گیری کرد و خطوط را کشید یا رنگ کرد (رجوع شود به توضیحات سرور – صفحه بازی).

این کلاس تنها شامل سه متغیر صحیح با دسترسی عمومی با نام های row و column و side می باشند. یکی از کاربرد های این کلاس پیدا کردن خانه ها و اضلاع خالی در صفحه بازی است.

زمین بازی – آرایه Board

این آرایه به صورت زیر در برنامه تعریف شده است:

```
int Board[15][15][5] = {0};          //Game board
```

این آرایه وظیفه نگه داری اطلاعات صفحه بازی را بر عهده دارد. همان طور که مشاهده می کنید این آرایه در سایز 15x15x5 تعریف شده چرا که حداکثر تعداد سطر ها و ستون ها برای اجرای بازی ۱۵ سطر و ۱۵ ستون است. به علاوه هر خانه دارای ۴ ضلع و یک متغیر دیگر است که مشخص کننده وضعیت آن خانه می باشد. لذا در بعد سوم این آرایه (برای هر خانه) ۵ عنصر تعریف شده است.

به عنوان مثال سطر r و ستون c را در نظر بگیرید در این صورت برای دسترسی به وضعیت خانه (رنگ شده یا رنگ نشده بودن) موجود در این سطر و ستون باید محتویات Board[r][c][0] را بررسی کنیم، برای دسترسی به وضعیت ضلع سمت چپ از خانه موجود در سطر r و ستون c باید Board[r][c][1] را بررسی کنیم، برای دسترسی به وضعیت ضلع بالا از خانه موجود در سطر r و ستون c باید Board[r][c][2] را بررسی کنیم، برای دسترسی به وضعیت ضلع سمت راست از خانه موجود در سطر r و ستون c باید Board[r][c][3] را بررسی کنیم، برای دسترسی به وضعیت ضلع پایین از خانه موجود در سطر r و ستون c باید Board[r][c][4] را بررسی کنیم.

توجه کنید در صورتی که وضعیت خانه ای مشخص نشده باشد یا اینکه در ضلعی از خانه ای هنوز خطی کشیده نشده باشد، مقدار متناظر با آن در Board برابر با صفر خواهد بود.

متغیر های سراسری

نام متغیر	مورد استفاده	تعریف متغیر در برنامه
ROW	مشخص کننده تعداد سطر های زمین بازی در بازی فعلی	`int ROW = 0;`
COLUMN	مشخص کننده تعداد ستون های زمین بازی در بازی فعلی	`int COLUMN = 0;`
MyTurn	مشخص کننده شماره نوبت برنامه شما	`int MyTurn = 0;`
HisTurn	مشخص کننده شماره نوبت برنامه حریف	`int HisTurn = 0;`

نکته مهم: تعداد سطر و ستون ها بر اساس تعداد نقاط به عامل فرستاده میشوند، توجه کنید که تعداد خانه ها در سطرها و ستون ها، همیشه یک واحد کمتر از تعداد نقاط است و آرایه Board بر اساس تعداد خانه ها عمل می کند (رجوع شود به شکل صفحه اول و همچنین کد نمونه در تابع Decide)

تابع Initialize

تعریف این تابع به صورت زیر است:

```
void Initialize();  //Initialize essential variables
```

همان طور که مشاهده می کنید این تابع هیچ پارامتری دریافت نمیکند و تنها یک بار در تابع main() فراخوانی شده است. وظیفه این تابع دریافت اطلاعات اولیه بازی (مشخصات بازی) است. این مشخصات به ترتیب شامل تعداد سطر ها، ستون ها و نوبت بازیکن فعلی است که از ورودی استاندارد خوانده شده و در متغیر های ROW، COLUMN و MyTurn جایگذاری می شود.

نوبت بازیکن با یکی از اعداد ۱ یا ۲ مشخص می گردد. به عبارت دیگر اگر نوبت ما ۱ باشد ما شروع کننده بازی خواهیم بود و در تمامی دور های با شماره ۱، تصمیم گیری با برنامه ما خواهد بود و به تبع آن نوبت برنامه مقابل ۲ بوده و در تمامی دور های با شماره ۲، تصمیم گیری خواهد نمود. لذا این تابع متغیر HisTurn را نیز که مخصوص بازیکن مقابل است با توجه به MyTurn مقدار می دهد.

تابع CheckRectangle

تعریف این تابع به صورت زیر است:

```
void CheckRectangle(int row,int column,int turn)
```

این تابع برای آن است که ببینیم آیا مستطیلی که در سطر row و ستون column قرار دارد کامل شده است؟ در صورت مثبت بودن جواب به آن مستطیل (خانه board[row] [column] [0]) مقدار turn نسبت داده می شود.

تابع CheckForAchivements

تعریف این تابع به صورت زیر است:

```
void CheckForAchivements(int row,int column,int side,int turn)
```

این تابع زمانی فراخوانی می شود که خط جدیدی کشیده شده باشد (به عبارت دیگر ضلعی رنگ شده باشد) در این صورت ابتدا خانه های متناسب با آن ضلع چک می شوند که آیا تکمیل شده اند یا خیر (با استفاده از تابع CheckRectangle). توجه کنید که هر ضلع بین دو خانه مشترک می باشد و با پر شدن آن ممکن است دو خانه همسایه کامل شوند.

تابع UpdateBoard

تعریف این تابع به صورت زیر است:

```
void UpdateBoard(int row,int column,int side,int turn)
```

این تابع ابتدا ضلع واقع شده در سطر row، ستون column و ضلع side از Board را برابر turn قرار می دهد تا مشخص شود کدام بازیکن این حرکت را انجام داده است (از وضعیت ۰ به یکی از مقادیر ۱ یا ۲ با توجه به مقدار ذخیره شده در turn ارتقا یابد).

```
Board[row][column][side] = turn;
```

توجه داشته باشید که هر خانه با ۴ خانه دیگر همسایه است، لذا در صورت کشیده شدن یک ضلع، در واقع دو درایه از ماتریس Board پر خواهد شد.

تابع UpdateBoard وظیفه یافتن سلول همتا با سلول سطر row ، ستون column و ضلع side را دارد، که آن را نیز با مقدار turn مقدار دهی می نماید.

تابع Decide

تعریف این تابع به صورت زیر است:

```
void Decide()
```

این تابع وظیفه تصمیم گیری برای حرکت فعلی را دارد. دستورات این تابع باید توسط شما نوشته شود تا برای حرکت فعلی تصمیم گیری نمایید.

به عنوان مثال اگر برنامه نوشته شده توسط شما سلول سطر r ، ستون c و ضلع s را انتخاب نمود باید این مقادیر را به ترتیب در خروجی استاندارد چاپ کند. توجه کنید مقادیر r، c و s باید به همین ترتیب و بدون هیچ مقدار اضافه ای چاپ شوند و بین آنها یک فضای خالی (فاصله یا space) وجود داشته باشد، و پس از چاپ کردن این مقادیر می بایست یک '\n' یا endl نیز چاپ کنید. در غیر این صورت دستورات غیر معتبر تلقی شده و پردازش نمیشوند. (رجوع شود به توضیحات سرور - قوانین و خطاها) در نهایت باید تابع UpdateBoard را به ترتیب با پارامتر های r، c، s و MyTurn فراخوانی کنید (در غیر این صورت Board در برنامه شما به روز رسانی نخواهد شد).

در تابع Decide برنامه ای را به عنوان نمونه نوشته ایم که در اینجا به توضیح آن خواهیم پرداخت.

ابتدا با دستور

```
vector<Cell> EmptyCells;
```

یک vector از نوع Cell به نام EmptyCells ساخته شده است. (لازم به یاد آوری است vector نوعی ساختمان داده است شبیه آرایه که می توان در هر لحظه عنصری به انتهای آن اضافه نمود – برای کسب اطلاعات بیشتر به سایت رسمی مسابقات مراجعه کنید).

در این vector تمام سلول های خالی را قرار خواهیم داد، بدین منظور ابتدا می بایست Board را پیمایش کنیم و سلول هایی که هنوز مقدار آنها صفر است را بیابیم. برای این کار از سه حلقه for تو در تو و یک if به صورت زیر استفاده مینماییم:

```
for(int i=0;i<ROW-1;i++)
{
	for(int j=0;j<COLUMN-1;j++)
	{
		for(int k=1;k<5;k++)
		{
			if(Board[i][j][k] == 0)
			{
				Cell c;
				c.Row = i; c.Column = j; c.Side = k;
				EmptyCells.push_back(c);
			}
		}
	}
}
```

همانطور که قبلا گفته شد، اعداد داخل ROW و COLUMN بر حسب تعداد نقاط می‌باشند اما ابعاد آرایه زمین، هم در کد عامل و هم در سرور، بر اساس تعداد خانه‌ها تعیین می‌شوند. در نتیجه در ارسال دستورها و پیمایش آرایه، این نکته را رعایت کنید.

پس از آنکه تمامی سلول های خالی پیدا شدند و در EmptyCells قرار گرفتند یکی از آنها را با دستور زیر به تصادف انتخاب کرده و مقدار آن را در شی selected که از نوع Cell تعریف شده است کپی می کنیم.

```
Cell selected = EmptyCells[rand()%EmptyCells.size()];
```

در نهایت با استفاده از دستور زیر شماره سطر، ستون و ضلع سلول انتخابی را با قالب توضیح داده شده در خروجی چاپ می کنیم. و تابع UpdateBoard را فرا میخوانیم:

```
//Print out agent decision
cout << selected.Row << " " << selected.Column << " " <<
selected.Side << endl;

//Updating board according to your agent decision
UpdateBoard(selected.Row,selected.Column,selected.Side,MyTurn);
```

تابع main

در ابتدای تابع main() مشاهده می شود که با استفاده از تابع srand به random seed مقدار time(0) داده شده، دلیل آن استفاده از اعداد تصادفی در برنامه نمونه نوشته شده در تابع Decide() برای انتخاب تصادفی یکی از سلول ها می باشد.

سپس تابع Intialize فراخوانی شده تا به متغیر های سراسری مقدار اولیه داده شود، در نهایت یک حلقه بی نهایت ایجاد شده است.

در این حلقه سه متغیر صحیح با نام های row ، column و side تعریف شده و سپس از ورودی استاندارد خوانده شده اند. با توجه به توضیحات سرور که اگر مقدار ورودی ها 1- 1- 1- بود باید تصمیم بگیریم و تابع Decide() فراخوانی می شود و در صورتی که ورودی ها غیر از این مقدار بود، در حقیقت سرور تصمیم برنامه حریف را به اطلاع ما می رساند، و برنامه با استفاده از تابع UpdateBoard و با توجه به مقادیر خوانده شده عملیات به روز رسانی زمین بازی (آرایه Board) را انجام می دهد.

سخن پایانی

از آنجا که وجود خطا های منطقی (Bug) در هر نرم افزاری اجتناب ناپذیر است، ممکن است در این برنامه نیز خطاهایی از این دست یافت شود لذا خواهشمند است در صورتی که خطایی یافتید در اسرع وقت آن را از طریق آدرس ایمیل robotics@shahed.ac.ir به کمیته برگزاری مسابقات گزارش فرمایید، تا نسبت به رفع آن، اقدامات لازم صورت پذیرد.