Game Teory ,میکروپروسسور , میکروکنترله

مقدمه :

در حال حاضر رشتهٔ برق الکترونیک از حد یک رشته فراتر رفته و پای در عرصه های گوناگونی گذاشته است . کمتر رشته ای را می توان یافت که از برق الکترونیک تاثیر نپذیرفته باشد امروز بشر در هر رشته ، فن و صنعتی پا بگذارد نه تنها از این رشته بی نیاز نیست بلکه برای موفقیت بیشتر کاملاً به آن نیازمند است به گونه ای که کشورهایی که در استفاده از آن سرعت عمل به خرج نمی دهند در بسیاری از زمینه ها دچار عقب ماندگی ها و ضررهای جبران ناپذیری می شوند .

رشتهٔ برق الکترونیک امروزه در صنعت ، پزشکی ، هنر، علوم تربیتی ، علوم کامپیوتر، ارتباطات، اقتصاد، کشاورزی، تجاری و علوم نظامی به یکی از مهمترین مولفه ها تبدیل شده است این رشته به گونه ای وسیع است که در بسیاری از زمینه ها رشته های جدیدی بین این رشته و رشته های دیگر ایجاد شده است . (مهندسی پزشکی، مکاترونیک ، جنگ الکترونیک، ...)

بنابراین بحث الکترونیک در عرصه ای مانند Game نیز بدون حسن نبوده و در بسیاری از موارد جنبهٔ اقتصادی و آموزشی دارد. چنانچه بسیاری از Game ها با جنبهٔ آموزشی که با واقعیت خارجی منطبق اند خود به تنهایی آموزش نیروی انسانی را به عهده می گیرد برای مثال Game هایی که کاربر با نشستن در محیطی شبیه اتاق کنترل هواپیما تحت آموزش مجازی قرار می گیرد و با نمونه های دیگری در آموزش عمل جراحی و نظایر این نوع بازی ها.

هدف:

در پروژه فعلی انجام شده دو ویژگی خاص وجود دارد یکی وارد کردن الکترونیک به مبحث Game Teory می باشد که یکی از نظریه های بسیار مهم بوده و موارد استفادهٔ زیادی در مباحث اقتصادی ، تجاری، سیاسی و نظامی دارد. این پروژه با وارد کردن این مبحث در رشتهٔ الکترونیک و طراحی میکروپروسسورها کار جدیدی در این مورد به حساب می آید همچنین قدرت میکروکنترله های AVR در وارد شدن به چنین مباحثی را نشان می دهد ثانیاً در این پروژه امکان تعریف حرکات جدید برای مهره ها وجود دارد که هدف از این قسمت پروژه نشان دادن مقدار انعطاف پذیری الکترونیک و طراحی میکروکنترلر های AVR می باشد چرا که در این پروژه، طراحی به گونه ای انجام شده است که سیستم بتواند تمامی کارهایی را که برای حالت عادی می توان انجام داد برای هر حالت جدید دیگری که خود کاربر تعریف می کند نیز بتواند انجام دهد در واقع حالت عادی فقط یک حالت خاص از تعداد بی شمار حالتی است که کاربر می تواند تعریف کند و این انعطاف در طراحی میکروکنترلر ها می باشد.

فصل اول

مستندات پروژه:

تئوری بازی :

Game teory با استراتژی های رقابتی یک نظریه ریاضی است که شامل تصمیم گیری در شرایط تعارض بوده و با موقعیت های رقابتی سروکار دارد در این نظریه تصمیم گیرنده با توجه به استراتژی های رقیب عملکرد خود را مورد ارزیابی قرار می دهد در چنین موقعیتی تصمیم یکی از بازیکنان بر تصمیم گیری سایرین تاثیر می گذارد. این نظریه در زمینه های بسیار وسیعی کاربرد دارد مانند پیکار دو بازیکن در بازی شطرنج مبارزات سیاسی در انتخابات، طرحهای عملیات جنگی و شرایط رقابت شرکت های تجاری در بازار

و غیره. اساساً می توان گفت این نظریه توسط ون نیومن ارائه شد، او اصل مینیماکس (کمیته بیشینه) را ارائه نمود . طبق این اصل هر رقیب طوری عمل می کند که حداکثر زیان (باخت) خود را حداقل نماید و یا حداقل سود (برد) خود را به حداکثر برساند.

هر چند نظریهٔ بازیها در سال 1928 توسط ون نیومن مطرح گردید، وی از سال 1944 به بعد با انتشار کتاب « نظریهٔ بازی ها و رفتار اقتصادی » که توسط او و مورگانسترن، مورد توجه شایانی قرار گرفت . و با توسعه کاربردهای آن در اقتصاد و جایگاه مهمی پیدا کرد.

تاریخچهٔ نظریهٔ بازی را اصولاً می توان در منابع مذهبی و اسلامی پیدا کرد و از کاربردهای آن در اینگونه منابع نیز مطالبی ارائه نمود. برای مثال نمونه های آن را در کتاب قضاوت های حضرت امیرالمومنین علی (ع) می توان یافت . در نامه ای که در تاریخ 13 نوامبر 1713 میلادی : جیمز والگریو نوشت اولین راه حل استراتژی مینی ماکس مخلوط به یک بازی دو نفره مطرح شد . والگریو در نامه اش در بارهٔ یک بازی دو نفره که قبلاً نیکولاس برنولی مطرح کرده بود، مطالبی نوشت هر چند راه حل پیشنهادی والگریو یک معادله مخلوط کمینه پیشینه بود . ولی اوسایر بازیها را توسعه نداد و روی بازیهای دارای استراتژی مخلوط و احتمالی متمرکز شد و قوانین معمول بازیها در آن به چشم نمی خورد. در سال 1913 میلادی برای اولین بار در مورد بازی شطرنج عنوان شد که « شطرنج تنها یک مجموعه برنده معقول و منحصر به فرد در استراتژیهای محض دارد » این قضیه بوسیله ی زرملو در مقاله اش منتشر شد.

در سال 1928 میلادی جان ون نیومن قضیهٔ کمینه بیشینه را در مقاله اش اثبات کرد. او بیان کرد که بازی دو نفره مجموع صفر با استراتژی های زیاد و محدود برای هر بازیکن معین است . این یک بازی متغیر با بردار سود معقول و منحصر به فرد بود که با استفاده از بعضی محاسبات تابعی و توپولوژی اثبات می گردید در این مقاله یک بازی به شکل گسترده (extensive form) نیز مطرح گردید.

طی چهارمقاله بین سالهای 1950 و 1953 جان نش روی نظریه بازی های بدون همکاری و نظریهٔ چانه زنی مطالبی مطرح کرد . در دو مقاله نقاط تعادل در بازی های دونفره (1950) و بازی های بدون همکاری (1951) مطرح شد . نش وجود یک منطقه استراتژیک را برای بازیهای بدون همکاری تحت عنوان – منطقه ی تعادل نش – و طرح بازی های بدون همکاری را اثبات کرد.

بازیهای دیفرانسیل ای بوسیلهٔ رافوس ایکاس در 1950 توسعه داده شد. آنها مسئله را رشد دادند و بازیهای نظامی تعقیب و گریز را شکل دادند. اولین انتشارات در این حوزه در باره بازیهای دیفرانسیل ای یادداشت تحقیقی شرکت رانه بود که در 25 مارس 1955 منتشر شد . در سال 1962 میلادی یک مورد استفاده از نظریه بازی در بیمه در مقاله کارل بورچ تحت عنوان « کاربرد نظریه بازی برای بعضی از مسائل بیمه خودرو» مطرح شد. مقاله مشخص کرد که چگونه نظریه بازی میتواند زمانی که حق بیمه برای همه انواع مختلف بیمه کار، استخراج می شود . بکار گرفته شود. بورچ اثبات کرد که عدد شاپلی حق بیمه معقولی را برای انواع ریسک ارائه می دهد.

در سال 1972 میلادی مفهوم استراتژی پایدار تکاملی (ESS) برای نظریه بازی بوسیلهٔ جان مینارد در یک مقاله تحت عنوان « نظریه بازی و تکامل جنگ» مطرح شد. استفاده از مفهوم ESS در اقتصاد و بیولوژی گسترش یافت. در سال 1992 میلادی کتابی تحت عنوان « نظریهٔ بازی با کاربردهای اقتصادی، جلداول» بوسیلهٔ رابرت اومان و سرگئی هارت انتشار یافت . در سال 1994، « نظریهٔ بازی و قانون » بوسیلهٔ دالگلاس برد و رابرت گرتز و راندال پیکر که یکی از اولین کتاب ها در اقتصاد و قانون است و یک روش صریح برای بدست آوردن جواب نظریه بازی ارائه می کند، منتشر شد.

کتاب « نظریه ی بازی با کاربردهای اقتصادی جلدی 2» بوسیلهٔ اومان و سرگئی هارت در سال 1994 انتشار یافت ، در همین سال جایزهٔ بانک مرکزی سوئد در علم اقتصاد به یاد بود آلفرد نوبل به جان نش و جان ها راسینی و دینهارد سلتن برای فعالیت های آنها در زمینهٔ نظریهٔ بازی اعطا شد در سال 2002 میلادی جایزه مرکزی سوئد در علم اقتصاد به یاد بود الفرد نوبل به ورنان اسمیت و رابرت اومان و توماس شلینگ برای فعالیت های آنها در زمینهٔ نظریه باز ی اعطا شد.

در نظریهٔ بازی برای یک بازی؛ مشخصه های زیر مطرح می شود:

1- تعداد شرکت کنندگان یا رقبا محدود است و تعداد شرکت کنند گان در یک بازی که از این پس بازیکن نامیده می شوند حداقل باید دو نفر باشند تا بازی انجام شود . و در این حالت بازی را دو نفره می گویند.

2- هر بازیکن فهرست محدودی از فعالیت های امکان پذیر و اجرا شدنی را دراختیار دارد که ممکن است این فهرست برای بازیکنان مختلف متفاوت باشد.

3- هر بازیکن ممکن است از انتخاب های احتمالی سایر بازیکنان آگاهی داشته باشد. اما او نمی داند سایرین چه تصمیمی را اتخاذ نموده اند.

4- هر ترکیبی از انتخاب استراتژیها تعیین کننده ی پیامدی است که برای بازیکنان سود(یا ضرر) به همراه دارد.

5- سود یا ضرر یک بازیکن به انتخاب های سایر بازیکنان یا رقبای او نیز بستگی دارد.

حال می توانیم بعضی از مفاهیمی را که در تئوری بازی کاربرد زیادی دارند را تعریف کنیم.

بازی: فعالیتی است میان دو یا چند نفر شامل انتخاب استراتژی های مختلف و اجرای آنها طبق مجموعه ای از قواعد مورد اتفاق ، که برای هر کدام از بازیکنان سود یا ضرر به همراه دارد. اگر در یک بازی انتخاب استراتژی ها به صورت تصادفی انجام شود آن را بازی تصادفی و درغیر این صورت آن را بازی استراتژیک می نامند .

بازیکن: هر شرکت کننده یا رقیب که در بازی مجاز به انتخاب استراتژی می باشد بازیکن نامیده می شود.

استراتژی: مجموعهٔ فعالیت های پی در پی که از پیش تعیین شده و بازیکن طبق آن عمل می کند.

استراتژی خالص: قاعدهٔ تصمیم گیری برای انتخاب همیشگی یک حرکت خاص است که بازیکن تنها مجاز به انتخاب همان قاعده می باشد همانند حرکت رو به جلو در بازی منچ.

استراتژی مخلوط: قاعده تصمیم گیری برای انتخاب ترکیبی از استراتژی های مختلف و ممکن است که بازیکن براساس نوعی توزیع احتمال، آن را انجام می دهد . بنابراین استراتژی مخلوط در حقیقت انتخابی است که با احتمالات مختلف میان استراتژی های خالص صورت می گیرد.

بازی ایستا: اگر بازی دارای قوانین ثابت ای باشد بازی ایستا نامیده می شود.

با توجه به مطالب گفته شده تئوری بازی را برای یک بازی ایستای دو نفره به این صورت می توان توضیح داد:

ابتدا کاری که باید یک بازیکن انجام دهد این است که با توجه به قوانین بازی و با توجه به امکانات ای که بازیکن دارد یکسری احتمالات را برای انجام حرکات ایجاد کند . در ایجاد احتمالات بایستی هر کمیت را در نظر گرفت و هم کیفیت را، به عبارت دیگر به هر اندازه تعداد احتمالات بیشتر باشد به همان اندازه نسبت احتمالات با ارزش تر بیشتر خواهد شد همچنین بایستی سعی در ایجاد بهترین احتمالات کرد تا هنگام انجام حرکت امتیاز بیشتری گرفت . استفاده ای که ما از این قسمت در بازی شطرنج کرده ایم مربوط

به زیر برنامهٔ pos- Rate می باشد که حرکت هایی را به عنوان حرکت خوب انتخاب می کنیم که در مراحل بعد بیشترین حرکات ممکن را داشته باشیم در ضمن بیشترین دفاع را از مدافعان انجام داده و بیشترین فشار را بتوانیم به مهره های حریف وارد کنیم . درواقع با این کار یک سری احتمالات را با استفاده از قوانین شطرنج و بااستفاده از فرصتی که در اختیار مان قرار می گیرد (نوبت بازی) ایجاد کرده ایم که در ایجاد این احتمالات کمیت و کیفیت مهم می باشد. حال قسمت دیگر Game teory به این صورت است که : با انجام یک حرکت حریف حرکتی انجام می دهد و سپس نوبت به بازیکن اول و حرکت او و سپس به بازیکن دوم می رسد انجام حرکت ها تا زمانی ادامه می یابد که منجر به گرفتن امتیازی برای طرفین شود . برای توضیح این مطلب فرض می کنیم که با انجام یک حرکت و سپس انجام حرکت توسط حریف امتیازی گرفته می شود و یا اینکه امتیازی از دست می رود.

بنابراین در این صورت برای هر حرکت بازیکن اول ممکن است حرکت های مختلفی برای بازیکن دوم به وجود آید و در نتیجه برای هر حرکت بازیکن اول تعدادی از امتیازها به وجود می آید . برای مثال فرض کنیم جدولی مانند جدول رو به رو را داریم که در آن بازیکن اول حرکت های a تا d را دارد و بازیکن دوم حرکت های `a تا `d را می تواند داشته باشد . و امتیاز بازیکن اول بعد از انجام حرکت بازیکن اول و حرکت بازیکن دوم کسب می شود. طبق نظریه بازی چون حرکت بازیکن دوم همواره به گونه ای است که بیشترین امتیاز را گرفته و کمترین امتیاز را بدهد بنابراین امتیازی که بازیکن اول از هر حرکت می گیرد برابر کمترین امتیاز ممکن می باشد بنابراین امتیاز بازیکن اول در جدول بالا برای حرکت a ، 3 - می باشد و برای حرکت b ، یک ، برای حرکت C بازیکن دوم حرکتی ندارد (همانند پات شدن یا مات شدن در بازی شطرنج) و بازی تمام شده است و برای هر حرکت d برابر 1- است . بنابراین برای بازیکن اول حرکت a با اینکه ممکن است امتیاز 5 را هم به همراه داشته باشد ولی با این حال حرکت مناسبی نیست و حرکت b حرکتی است که برای بازیکن اول حداقل یک امتیاز به همراه دارد و حرکت C بازی را به نفع بازیکن اول تمام کند (همانند مات کردن حریف) این حرکت بهترین حرکت است ولی اگر این حرکت به نفع بازیکن اول نیست بهترین حرکت حرکت b خواهد بود. این جدول حالت دوبعدی بازی را نشان می دهد یعنی درد و حرکت امتیاز حرکت ها مشخص می شود ولی در بعضی از موارد می تواند چند بعدی باشد یعنی امتیازها بعد از چندین حرکت مشخص می شود. در هر حال بهترین حرکت، حرکتی است که بیشترین امتیاز را از بین کمترین امتیازهایی که حریف داده است نتیجه دهد.

بازیکن دوم

بازیکن اول

	a`	b`	c`	d`
a	5	0	-	-3
b	1	3	2	2
c	-	-	-	-
d	-1	0	1	-

استفاده ای که از این نظریه در بازی شطرنج می توان کرد این است که ابتدا تمامی امتیازاتی که یک ـر ـب می تواند داشته باشد را پیدا کرده سپس کمترین امتیاز را به عنوان امتیاز این حرکت انتخاب می کنیم سپس برای حرکت های بعدی نیز به همین ترتیب عمل کرده و پس از آن حرکتی که بیشترین امتیاز را دارد به عنوان بهترین حرکت انتخاب می شود. یافتن کمترین ارزش اکتسابی یک حرکت را در برنامهٔ بازی شطرنج ، زیر برنامهٔ Time-Rate انجام می دهد و یافتن حرکتی که بیشترین امتیاز را بدست می آورد به عهدهٔ زیر برنامهٔ Moreselection است .

اصول اساسی بازی شطرنج :

فصل دوم – سخت افزار:

فصل سوم - الگوریتم ها و منطق بازی:

فصل چهارم – برنامه وکدهای پروژه :

برای برنامه ریزی میکروپروسسورهای سری AVR انواع متنوعی از کامپایلرها عرضه شده است که مهمترین آنها Bascom ، Codevision و FAST AVR می باشد کامپایلری که برای کد نویسی این پروژه انتخاب شده است Bascom می باشد که این کامپایلر تمامی میکروکنترلرهای AVR را حمایت کرده و از زبان BASIC برای برنامه نویسی AVR ها استفاده می نماید از قابلیت های بسیار ارزنده محیط Bascom داشتن تحلیل گر یا به عبارتی Simulator داخلی است .

ورودی سیگنال آنالوگ ADC و مقابله کننده آنالوگ، ایجاد پالس بر روی پایه ای خاص، صفحه کلید 4 * 4 ، LCD ، ایجاد تمام وقفه ها به صورت اختیاری نوشتن ، خواندن حافظه EEPROM و SRAM ، رویت تمام رجیسترها و متغیرهای محلی و سراسری برنامه، اجرای برنامه به صورت خط به خط ، رویت صفر یا یک بودن تمام پایه توسط LED ، تغییر منطق پایهٔ دلخواه و بسیاری از امکانات دیگر توسط محیط تحلیل گر (Simulator) باعث شده است که در این پروژه از این کامپایلر استفاده کنیم.

برنامه از چهار قسمت تشکیل شده است که عبارتند از : 1- تعریف سخت افزار 2- تعریف متغیرها 3- معرفی زیر برنامه ها 4- برنامهٔ اصلی و زیربرنامه ها

اساس کار این برنامه به این صورت است که تعدادی تصویر گرافیکی در برنامهٔ paint ویندوز طراحی کرده و سپس توسط منوی Tools و قسمت Geraphic Converter تبدیل به فایل Bgf . می کنیم و در کنار برنامهٔ Bascom ذخیره می گردد . این تصاویر گرافیکی شامل: خانهٔ سفید خالی، خانهٔ سیاه خالی ، مهره های سفید با زمینهٔ سفید، مهره های سیاه با زمینهٔ سفید ، مهره های سفید با زمینهٔ سیاه و مهره های سیاه با زمینهٔ سیاه می باشد . که کلاً 26 تصویر گرافیکی داریم که در زیر برنامهٔ Show Picture نمایش داده می شود . زیر برنامهٔ Show Picture به این صورت عمل می کند که متغیری به نام Phi را داریم که مشخص کنندهٔ کد تصویر است . Phi می تواند مقادیر 0 تا 25 را داشته باشد . زیر برنامهٔ Show Picture با توجه به مقدار Phi و محل نمایش که x و y می باشند تصویر را بر روی LCD گرافیکی و در طول x و عرض y نمایش می دهد . دستور نمایش تصویر در این زیر برنامه به این صورت است :

Show Pic x,y , Label

این دستور تصویری را که در Label مشخص شده است را در طول x و عرض y نمایش می دهد .

Label به صورت زیر است :

```
Label:
$ bgf    ``نام تصویر Bgf``
```

در این برنامه کد تصاویر به صورت زیر است .

0 : خانهٔ سفید خالی

1: سرباز سفید با زمینهٔ سفید

2: قلعهٔ سفید با زمینهٔ سفید

3: اسب سفید با زمینهٔ سفید

4: فیل سفید با زمینهٔ سفید

5: شاه سفید با زمینهٔ سفید

6: وزیر سفید با زمینهٔ سفید

7: سرباز سیاه با زمینهٔ سفید

8: قلعهٔ سیاه با زمینهٔ سفید

9 : اسب سیاه با زمینهٔ سفید

10: فیل سیاه با زمینهٔ سفید

11: وزیر سیاه با زمینهٔ سفید

12: شاه سیاه با زمینهٔ سفید

13: خانهٔ سیاه خالی

14: سرباز سفید با زمینهٔ سیاه

.

.

.

.

.

.

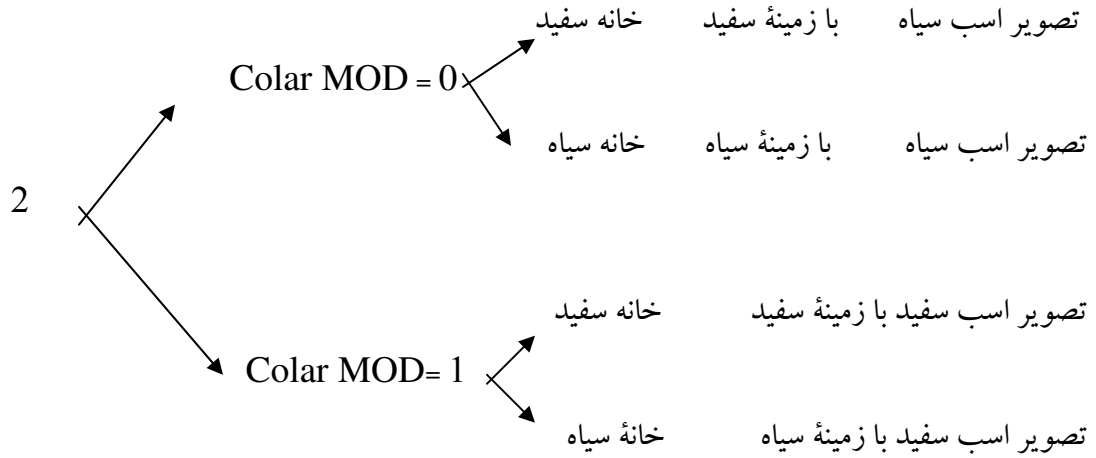
25: شاه سیاه با زمینهٔ سیاه

زیر برنامهٔ Show Picture با توجه به مقدار Pni یکی از 26 تصویر را در طول x و عرض y نمایش می دهد کد تصاویر با Pni از روی کد مهرها و رنگ مهره های انتخابی کاربر و نیز رنگ خانه ای که مهره در آن قرار می گیرد بدست می آید.

برای نمایش تصاویر بر روی صفحهٔ 64 خانه ای شطرنج ما متغیری 64 بایتی تعریف کرده ایم که Page نام دارد و هر خانهٔ آن معرف 64 خانهٔ صفحهٔ شطرنج می باشد . در این 64 خانه و در هر بایت آن یک کد قرار می گیرد این کدها در واقع همان مهره هایی هستند که در روی صفحهٔ شطرنج قرار می گیرد . این کدها 33 عدد بوده و عبارتند از :

0- خانهٔ خالی بدون مهره 1- قلعهٔ CPU 2- اسب CPU 3- فیل CPU 4-وزیر CPU 5- شاه CPU 6- فیل CPU 7- اسب CPU 8- قلعهٔ CPU 9 تا 16- سرباز CPU 17تا 24- سرباز کاربر 25- قلعهٔ کاربر 26- اسب کاربر 27- فیل کاربر 28- وزیر کاربر 29- شاه کاربر 30 فیل کاربر 31- اسب کاربر 32- قلعهٔ کاربر

این کدها همان مهره ها هستند ولی رنگ مهره ها و رنگ زمینهٔ تصویرشان مشخص نیست و به رنگ انتخابی کاربر و نیز رنگ خانه ای که مهره در آن قرار دارد بستگی دارد و برای مثال اگر کد مهره دو باشد چهار حالت برای کد تصویر وجود دارد . اگر کار بر رنگ سفید را انتخاب کرده باشد این تصویر، تصویر اسب سیاه خواهد بود و اگر این مهره در خانهٔ سفید قرار داشته باشد رنگ زمینهٔ این تصویر سفید خواهد بود وگرنه رنگ زمینه سیاه است . و اگر کاربر رنگ سیاه را انتخاب کرده باشد این تصویر، تصویر اسب سفید خواهد بود و رنگ زمینه نیز از روی رنگ خانه ای که مهره در آن قرار دارد مشخص می شود.



بنابراین باید زیر برنامه ای داشته باشیم که با توجه به نوع مهره و رنگ انتخابی کاربر و رنگ خانه ای که مهره در آن قرار دارد تصویر را بیابد . زیر برنامه ای که این کار را انجام می دهد Hame Picture نام دارد این زیر برنامه با توجه به مقدار Color MaD رنگ مهره ها و با Pn نوع مهره ها و نیز با توجه به رنگ خانه ای که مهره در آن قرار دارد رنگ زمینه را مشخص می کند و مقدار های 0 تا 25 را ایجاد می کند زیر برنامهٔ Hame Picture دو تا کار انجام می دهد . یکی یافتن کد تصویر و دیگری یافتن محل نمایش تصویر.

برای اینکه محل تصویر در روی LCP مشخص شود زیر برنامه محل کد را در متغیر Page پیدا می کند و با توجه به محل کد در متغیر Page محل تصویر را در روی LCD بدست می آورد . به این صورت که اگر کد را از بابت صفر متغیر Page محل تصویر را در روی LCP است نشان می دهد . و به همین ترتیب اگر کد از بابت یکم متغیر Page خوانده شود تصویر را در $x = 1 * 16$ و $y = 1 * 16$ نشان می دهد . برای پیدا کردن x و y از روی شمارهٔ بایت به این صورت عمل می کنیم: $y = Bni / 8$, $y = y * 16$ که Bni شمارهٔ بایتی است که کد از آن بایت خوانده شده است و $x = Bni - y * 8$, $x = x * 16$

برای پیدا کردن رنگ زمینه چون در سطرهای زوج صفحهٔ شطرنج خانه های زوج زمینهٔ سفید دارند و خانه های فرد زمینهٔ سیاه بنابراین کافی است زوج با فرد بودن شمارهٔ خانه را بررسی کرد و برای سطرهای فرد می توان عمل عکس را انجام داد.

الگوریتم Show Picture

Home relations : این زیر برنامه یکی از مهمترین قسمت های برنامه می باشد . اهمیت این زیر برنامه در این است که تمام روابط موجود بین مهره ها و خانه های دیگر صفحهٔ شطرنج را این زیر برنامه پیدا می کند . و نیز این زیر برنامه منعطف ترین قسمت برنامه می باشد به این دلیل که روابط بین یک مهره و خانه های صفحه شطرنج را برای هر نوع تعریف حرکت برای مهره پیدا می کند . یافتن رابطه بین مهره و خانه های صفحه شطرنج انجام می وشد. پس بنابراین کاری که این زیر برنامه باید انجام دهد این است که ابتدا نوع

مهره را پیدا کرده و از روی آن حرکات تعریف شده برای این مهره را بیابد . سپس با توجه به حرکات تعریف شده محل این مهره و مهره های دیگر رابطهٔ بین این مهره و خانه های دیگر صفحه شطرنج را بیابد.

این زیر برنامه شماره خانه را گرفته و سپس نوع مهره و تصویر آن را پیدا می کند . وبا توجه به نوع مهره حرکت تعریف شده برای آن را پیدا می کند . برای مثال برای مهره های با کد 9 تا 24 حرکت تعریف شده حرکت سرباز می باشد و نیز تصویر تعریف شده برای این مهره ها تصویر سرباز است . پس بنابراین اطلاعات ورودی برای این زیر برنامه شروع به پردازش حرکت های تعریف شده می کند . زیر برنامه همانگونه که قبلاً نیز گفته شده بایستی برای هر حرکت تعریف شده حرکت های متقارن دیگر را نیز پیدا کند به عبارت دیگر حرکت تعریف شده فقط برای ربع چهارم است آن هم در صورتی که حرکت تعریف شده با محل مهره جمع شود . بنابراین زیر برنامه از روی حرکت های تعریف شده حرکت های متقارن را در ربع های اول و دوم و سوم و چهارم را نیز می یابد.

پیدا کردن حرکت به این صورت است که چون حرکت برای حالت ای که مهره د رخانهٔ صفرام است تعریف شده بنابراین با تغییرجای مهره از خانهٔ صفرام به خانه های دیگر طول و عرض مکان مهره افزایش می یابد و چون مهره به طول و عرض جدیدی منتقل شده است حرکت های تعریف شده نیز به همان اندازه انتقال می یابند . مثلا اگر مهره به مکانی با طول x1 و عرض y1 منتقل شده باشد حرکت های تعریف شده برای این مهره نیز به اندازه x1 و y1 منتقل می شوند . حرکت مهره در ربع چهارم را ایجاد می کند حال برای پیدا کردن حرکت های متقارن بایستی ترکیبی از x2 و y2 و قرینه های آنها صورت گیرد x2 و y2 همان طول و عرض حرکت تعریف شده با x1 و y1 همان طول و عرض محل مهره است.

یعنی اگر طول و عرض حرکت تعریف شده با x1 و y1 جمع شود حرکت در ربع چهارم بدست می آید اگر طول و عرض حرکت تعریف شده از x1 و y1 جمع شود حرکت در ربع چهارم بدست می آید اگر طول و عرض حرکت تعریف شده از x1 و y1 جمع شود حرکت در ربع دوم بدست می آید . اگر با x1 کم شود و با y1 جمع گردد حرکت در ربع سوم بدست می آید.

برای مثال: اگر یکی از حرکت های تعریف شده برای مهرهٔ اسب خانهٔ شماره 10 باشد (طول2 و عرض1) اگر اسب به خانهٔ 35 (طول 3 و عرض 4) منتقل شود . حرکت در ربع اول برابر است با : (x = 3+2 =5 ، y = 4–1=3) ، خانهٔ شماره 27 (29= y*8+ x) . حرکت در ربع دوم برابر است : (x =3 –2=1، y =4 –1 =3)

خانهٔ شماره 25 (25 = y* 8+ x) . حرکت در ربع سوم برابر است با :

(x =3 –2=1، y =4 + 1 =5)، خانهٔ شماره ی 45 (45 = y* 8+ x) بنابراین با استفاده از حرکت تعریف شده بایستی چهار حرکت که متقارن هستند را پیدا کنیم. یافتن حرکت برای مهره ها با استفاده از حرکت های تعریف شده به ترتیب صورت می گیرد.

همنان طور که قبلاً نیز گفته شد متغیرهای king ، Queen، Bars، Elephant و Soldier حرکت های تعریف شده برای شاه ، وزیر، اسب ، فیل ، رخ و سرباز را در خود ذخیره می کنند . زیر برنامه با توجه به نوع مهره ابتدا مقادیر این متغیرها را در متغیر M قرار می دهد و سپس از بایت اول M شروع کرده و حرکت تعریف شده را می خواند . سپس حرکت های صحیح را برای مهره در هر ربع بدست آورده سپس این حرکت تعریف شده را می خواند . سپس حرکت های صحیح را برای مهره در هر ربع بدست آورده سپس این خانه را بررسی می کند تامشخص شود که چه مهره ای در این خانه وجود دارد . به همین ترتیب زیر برنامه بایت های دیگرM را نیز خوانده و حرکت های بعدی را بدست می آورد . حال نکته ای که اینجا وجود دارد این است که این حرکت ها دارای ترتیب می باشند یعنی اگر به بایتی برابر 64 برخورد نکنیم . حرکت های تعریف شده باید ترتیب را رعایت کنند یعنی اگر خانه ای

که در تعریف مقدم تر است دارای مهره باشد حرکت مهره به خانه های بعدی ممکن نیست ولی اگر بین این خانه و خانه های دیگر عدد 64 وجود داشته باشد ترتیب از بین می رود . برای مثال اگر مهرهٔ اسب دارای حرکت های 10 و 17 باشد اگر در خانه شمارهٔ 10 مهره ای وجود داشته باشد اسب از خانه صفرام نمی تواند به خانه 17 برود ولی اگر حرکت ها 10 و 64 و 17 باشد حرکت به خانهٔ 17 ارتباطی به بودن یا نبودن مهره در خانهٔ 10 ندارد و اسب می تواند به این خانه حرکت کند .

گفتیم که حرکتی که برای مهره تعریف شده فقط در ربع چهارم است و جا بایستی حرکت های متقارن دیگر را در ربع های اول و دوم و سوم نیز پیدا کنیم ولی باید این کار به گونه ای انجام شود که x و y خارج از صفحه نیفتند . یعنی x و y از صفر کوچکتر و از v بزرگتر نشود . همچنین در هر ربع در صورت عدم وجود عدد 64 در بین حرکات ترتیب وجود دارد ولی بین یک ربع و ربع دیگر ترتیبی وجود ندارد و حرکت های هر ربع از همدیگر مستقل اند.

پس بنابراین کاری که این زیر برنامه انجام می دهد این است که M را بایت بایت خوانده و در T قرار می دهد سپس آن را در Hr قرار داده و بررسی می شود. Hr سد حالت می تواند داشته باشد . اگر صفر باشد زیر برنامه تمام می شود . اگر برابر 64 باشد حرکت های بعدی وابستگی خود به حرکت های قبلی را از دست می دهند و اگر عددی بین یک تا 63 بود در این حالت این حرکت بررسی می شود. بررسی حرکت به این صورت است که قبل از این مرحله محل مهره (x1 و y1) پیدا شده است . و در این قسمت حرکت پیدا شده را به مولفه های (x2 ، y2) تبدیل می کند در ادامه هر یک از حالت های تقارن پیدا می شود. و اگر طول و عرض بین صفر و v بود و نیز تعداد مانع از دو تا کمتر بود. زیر برنامه ادامه پیدا می کند . سپس محل پیدا شده بررسی می شود. حال سه حالت پیش می آید. 1- مهره ای در این خانه وجود ندارد 2- مهره ای از CPU وجود دارد. 3- مهره ای از کاربر وجود دارد.

اگر مهره ای وجود نداشت f که مشخص کننده ی موانع است به همان مقدار قبل باقی می ماند و تغییری نمی کند اگر مهره از آن CPU بود F یک واحد افزایش می یابد و اگر کاربر بود 2 واحد افزایش پیدا می کند بررسی همواره تا F های کوچکتر از 3 ادامه دارد و اگر نه قطع خواهد شد با توجه به F که موانع را نشان می دهد و با توجه به مهره ای که در خانه ی پیدا شده وجود دارد 14 حالت مختلف برای کدهای به وجود می آید. که عبارتند از :

0- حرکت به این خانه تعریف شده است و یا بیش از دو مانع وجود دارد.

1- حرکت به این خانه تعریف شده است ومانع وجود ندارد و خانه خالی است .

2- حرکت به این خانه تعریف شده است و مانع وجود ندارد و خانه دارای مهره ی کاربر است .

3- حرکت به این خانه تعریف شده است و مانع وجود ندارد و خانه دارای مهره ی کاربر است .

4- حرکت به این خانه تعریف شده است و مانعی از مهره های CPU است و خانه خالی است.

5- حرکت به این خانه تعریف شده است و مانعی از مهره های CPU وجود دارد و خانه دارای مهره CPU است .

6- حرکت به این خانه تعریف شده است و مانعی از مهره های CPU وجود دارد و خانه دارای مهره ی کاربر است .

7- حرکت به این خانه تعریف شده است و مانعی از مهره های کاربر وجود دارد و خانه خالی است .

8- حرکت به این خانه تعریف شده است و مانعی از مهره های کاربر وجود دارد و خانه دارای مهره CPU است .

9- حرکت به این خانه تعریف شده است ومانعی از مهره های کاربر وجود دارد و خانه دارای مهره کاربر است .

10- مهره به این خانه فشار وارد می کند ولی نمی تواند به این خانه حرکت انجام دهد.

11- مهره به این خانه فشار وارد می کند ولی نمی تواند به این خانه حرکت انجام دهد.

12-حرکت به این خانه ممکن نیست . چون مانعی از مهره های CPU وجود دارد.

13- حرکت به این خانه ممکن نیست چون مانعی از مهره های کاربر وجود دارد.

حال زیر برنامه با فراخوانی Write code کد بدست آمده را در محل خود می نویسد و سپس با توجه به ربع ای که در آن قرار داشت F را در متغییرهای F1 تا F4 قرار می دهد . در MOD1 برای قلعه و وزیر ممکن است حالتهای تکراری پیش آید بنابراین ما خودمان از این حالتهای تکراری صرف نظر می کنیم تا سرعت پروازش زیاد باشد . و این کار به همین صورت ادامه پیدا می کند تا تمامی حرکت های ممکن برای مهره پیدا شود . برای مهره ی سرباز در حرکت دوم یافتن کد متفاوت است . چون حرکت دوم سرباز نشان دهنده ی بیشترین مقدار حرکت مضاعف است . بنابراین این حرکت یک خانه نیست که سرباز بتواند به آن حرکت کند بلکه یک مفهوم است و معنی آن این است که سرباز تا این خانه می تواند حرکت های مضاعف انجام دهد بنابراین باید به صورت دیگری عمل کرد به خاطر اینکه حرکت سرباز برای مهره های CPU و user بر عکس هم است بنابراین حرکت های user را باید معکوس کرد.

برای پیدا کردن حرکت سرباز اگر مال CPU باشد بایستی خانه ی جلوی سرباز را پیدا کرده و به عنوان حرکت قابل قبول انتخاب کرد و این کار زمانی قطع می شود که حرکت پیدا شده از حد نهایت حرکت مضاعف بیشتر شود و برای سرباز user برعکس این حالت پیش می آید یعنی خانه ی جلوی سرباز (y=y1-1) پیدا می شود و این کار زمانی قطع می شود که حرکت پیدا شده از معکوس حد نهایت حرکت مضاعف کمتر شود. برای مثال اگر حد نهایت حرکت مضاعف برای سرباز 24 (y =3) تعریف شده باشد ابتدا خانه ی جلوی سرباز پیدا شده سپس کد آن را نوشته پس از آن اگر این خانه از 24 بیشتر بود بررسی ادامه نمی یابد وگرنه ادامه پیدا می کند حال برای user 24 تبدیل به 32 می شود (y=3 y=7-y y=4) در این حالت اگر حرکت پیدا شده از معکوس حد نهایت حرکت مضاعف کمتر شد برنامه ادامه پیدا نمی کند و تمام خواهد شد .

همچنین اگر مانعی بر سر راه وجود داشت کدها برابر 12 خواهد بود یعنی مهره ی سرباز نمی تواند به این خانه ها حرکت کند.

با توجه به مطالب گفته شده ، این زیر برنامه با توجه به حرکت تعریف شده خانه هایی را که مهره می تواند به آنها حرکت کند را در همه ی حالتهای تقارن پیدا می کند با توجه به مهره های موجود در صفحه شطرنج و نیز موانعی که بر سر راه مهره وجود دارد کدهای صفر تا 13 را ایجاد می نماید همچنین پیدا کردن کدها تا زمانی ادامه می یابد که حرکت تعریف شده برابر صفر شود که در این صورت زیر برنامه تمام می شود و اگر حرکت تعریف شده برابر 64 باشد این عدد هیچکدام از خانه های صفحه شطرنج نیست و فقط نشان می دهد که حرکت های بعدی مستقل از حرکت های قبلی اند و مانعی بر سر راه این خانه وجود ندارد.

این زیر برناخمه به هنگام فراخوانی فقط شماره ی خانه را می گیرد و مهره ی آن را پیدا می کند و سپس کدها را برای این مهره پیدا می نماید بنابراین در طی یک فراخوانی فقط کدها را برای یک مهره بدست می آورد.

Move selection : تمامی زیر برنامه های قبلی مقدمه ای بود برای رسیدن به این مرحله که زیر برنامه و Move selection از آنها استفاده کند استفاده ای که Pointer از زیر برنامه های قبلی می کند توضیح داده شد، Pointer از Home relations برای تعیین صحت حرکت استفاده می کند و Move selection از این زیر برنامه برای تجزیه و تحلیل حرکت ها سود می برد . این زیر برنامه تجزیه و تحلیل متغیر 1024 بایتی Code- cup که اطلاعات آن را زیر برنامه Home relations تولید کرده است . و با

استفاده از Game teory حرکت صحیح را انتخاب می کند همانگونه که قبلا گفته شد در یک بازی ابتدا بایستی طبق قوانین بازی یکسری از احتمالات را ایجاد کرد و سپس از بین این احتمالات ، احتمالی را که بیشترین سود را برای ما داشته است را انتخاب نمود.

با توجه به این که در مقابل یک حرکت ما حریف چندین حرکت ممکن است داشته باشد بنابراین امتیازی که یک حرکت دارد بستگی به حرکت حریف دارد. بنابراین یک حرکت ما ممکن است نتایج گوناگونی داشته باشد .

برای مثال با توجه به جدول بازیکن اول سد حالت برای حرکت خود دارد و بازیکن دوم به ازای هر حرکت بازیکن اول تعدادی حرکت دیگر دارد. بنابراین هر یکاز احتمالات `aa تا `dd یک امتیاز حاصی خواهد داشت . اگر بازیکن اول حرکت a را انتخاب کند دو حالت پیش می آید و امتیازی که این حرکت بدست می دهد 1 و یا 3- خواهد بود با توجه به اینکه بازیکن دوم همواره حرکتی را انتخاب خواهد کرد که کمترین امتیاز را به ما بدهد بنابراین طبق نظریه ی بازی (Game teory) امتیازی که هر یک از حرکت های بازیکن اول می گیرد کمترین امتیاز ممکن خواهد بود . بنابراین کاری که این زیر برنامه انجام می دهد این است که ابتدا از بین امتیاز های که یک حرکت می تواند بگیرد کمترین آنها را انتخاب می کند یعنی امتیازی که از یک حرکت می توان بدست آورد کمترین امتیاز ممکن لحاظ می شود سپس از بین این حرکات ، حرکتی انتخاب خواهد شد که بیشترین امتیاز را نتیجه دهد.

	a`	b`	c`	d`
a	–	1	-	-3
b	0	1	0	–
c	5	7	10	1–
d	–2	–	–	0

بنابراین کاری که این زیر برنامه باید انجام دهد این است که ابتدا زیر برنامه حرکتی از CPU را انجام داد. سپس تمامی امتیازاتی را انتخاب می کند که این حرکت در بر دارد و در نهایت با مقایسه امتیاز این حرکت با حرکت قبلی، حرکتی که امتیاز بیشتری را بدست می آورد انتخاب می کند.

مابرای پیدا کردن امتیازی که یک حرکت می تواند داشته باشد سه نوع ارزش واگذاری انجام داده ایم که عبارتند از : 1- ارزش گذاری مهره ای 2- ارزش گذاری زمانی 3- ارزش گذاری موقعیتی .

ارزش گذاری مهره ای به این صورت است که هر یک از مهره ها دارای ارزش خاص خود می باشند که این ارزش گذاری با توجه به اهمیت ای است که مهره دارد برای مثال سرباز ارزش یک و وزیر ارزش 9 را دارد بنابراین گرفتن مهرهٔ وزیر دشمن ارزش بیشتری از سر باز خواهد داشت.

ارزش گذاری زمانی به این صورت است که ما ممکن است مهره ی کم ارزش را از حریف بگیریم ولی بعد از حرکت ما که نوبت به حریف می رسد حریف می تواند مهرهٔ با ارزش تری را از ما گرفته و در کل موفق تر عمل کند. به همین خاطر نیاز به یک ارزش گذاری زمانی می باشد که با توجه به نوبت بازیکن ها تمامی زد و خوردها را محاسبه کرده و ارزش گذاری می کند واز بین این ها امتیازهایی که از همه کمتر است را به عنوان امتیاز این حرکت انتخاب کرده و با ارزش مهره ای که cpu گرفته جمع می کند و در

نهایت همهٔ حرکت های بعدی را نیز بررسی کرده و امتیاز هر یک از حرکات را با حرکت های دیگر مقایسه و بیشترین را انتخاب می کند . نتیجتاً حرکتی را انتخاب می کند که بیشترین امتیاز را دارد.

در صورت تساوی امتیازها برنامه از ارزش گذاری موقعیتی استفاده می کند ، ارزش گذاری موقعیتی به این صورت است که از بین حالات ممکن حرکتی بهترین حرکت است که بیشترین فشار را به خانه های صفحهٔ شطرنج وارد کند و بیشترین دفاع از مهره های خودی را انجام دهد و بیشترین حمله ها به حریف را ایجاد نماید.

اگر در همآهٔ ارزش یابی ها تساوی برقرار باشد زیر برنامه از Random استفاده می کند .

حال با توجه به مطالب گفته شده توضیح الگوریتم این زیر برنامه به این صورت است که ابتدا متغیر Page که معرف صفحه شطرنج است ذخیره می شود تا در صورت جابه جایی مهره ها در طول این زیر برنامه بتوان صفحه را به حالت اول برگردانید سپس زیر برنامه از خانهٔ صفر شروع کرده و مهره های CPU را پیدا می کند و پس از آن حرکت های ممکن مهره ی پیدا شده را بدست آورده و سپس این حرکت را انجام می دهد در طی حرکت ، ممکن است دو حالت اتفاق بیفتد اول اینکه مهره بتواند در یک خانه ای که خالی است قرار بگیرد و دوم اینکه مهره بتواند یکی از مهره های حریف را بزند . در حالت دوم بایستی ارزیابی مهره ای انجام شود تا مقدار امتیازی که از این حرکت بدست می آید را پیدا کنیم . زیر برنامه چک می کند که پس از انجام حرکت آیا شاه خودی کیش شده است یا نه ، در صورتی که شاه کیش شده باشد حرکت انجام شده مورد قبول نبوده بنابراین صفحه ذخیره شده برگردانده می شود و سپس حرکت بعدی چک می گردد اولین حرکتی که پیدا می شود به عنوان بهترین حرکت انتخاب می شود در حرکت های بعدی مقایسه ها انجام می شود. ملاک انتخاب بهترین حرکت امتیازی است که این حرکت کسب می کند . برای ارزیابی ابتدا زیر برنامه های ارزیابی فراخوانی می شود . سپس اگر ارزش زمانی و مهره ای مساوی بودند ارزش موقعیتی مقایسه می شود در صورتی که حرکت فعلی از حرکت قبلی ارزش بیشتری داشت حرکت جدید به عنوان بهترین حرکت انتخاب می شود در غیر این صورت از Random استفاده می شود . اگر ارزش زمانی و مهره ای حرکت فعلی از حرکت قبلی بیشتر باشد حرکت فعلی به عنوان بهترین حرکت انتخاب می شود . سپس صفحهٔ ذخیره شده برگردانده شده کدها بازیابی می شود و حرکت بعدی بررسی می گردد این زیر برنامه اگر حرکتی پیدا کند که کار بر مات شود این حرکت را به عنوان بهترین حرکت انتخاب می کند. این عمل تا زمانی ادامه می یابد که همهٔ حرکت های ممکن انجام شده و تمامی خانه های صفحهٔ شطرنج بررسی شده باشد .

در این قسمت دو حالت وجود دارد، یا حرکتی پیدا شده است و بایستی این حرکت انجام شود و یا اینکه حرکتی پیدا نشده است . اگر هیچ حرکتی پیدا نشود در این صورت باز دو حالت پیش می آید یا شاه CPU کیش شده است و حرکتی وجود ندارد که در این صورت مات Flag یک می شود و بنابراین CPU مات است و یا CPU کیش نشده است که در این صورت بازی پات می باشد و پات Flag یک می گردد. در هر دو حالت پات بودن یا مات شدن زیر برنامه به قسمت End Game می رود و مات شدن یا پات بودن را اعلام می نماید.

با توجه به مطالب گفته شده در مورد زیر برنامه Move selection نشان می دهد که بایستی یک سری زیر برنامه وجود داشته باشد که عمل بررسی ، کیش بودن ، مات شدن ، ارزشیابی مهره ای ، ارزشیابی زمانی و ارزشیابی موقعیتی را انجام دهد . زیر برنامه هایی که این کار را انجام می دهد به ترتیب توضیح داده می شوند.

زیر برنامه Kish :

این زیر برنامه ابتدا محل ای را که شاه در آن قرار دارد را یافته سپس کد این محل را برای همهٔ مهره ها ی حریف بررسی می کند اگر حریف می تواند این خانه را هدف قرار دهد در این صورت شاه کیش است . اگر شاه کیش شده باشد کیش Flag یک می شود.

زیر برنامهٔ Pos- Rate :

این زیر برنامه با تجزیه و تحلیل اطلاعاتی که زیر برنامهٔ Home relations ایجاد می کند و در Code- cpu قرار می دهد با توجه به تعداد خانه هایی که مهره ها می توانند به آن حرکت کنند و یا تعداد خانه هایی که مهره ها می توانند به آن حمله کنند دست به ارزش گذاری این حرکت می زند . و دو تا مقدار ایجاد می کند یکی ارزش موقعیتی که CPU دارد و دیگری ارزش موقعیتی که user در این حرکت دارد. مقایسه این دو مقدار انجام نمی شود بلکه تفاضل این دو مقدار با تفاضل مقادیر قبلی مقایسه می شود. یعنی اگر حرکتی برای CPU ارزش موقعیتی برابر 20 ایجاد کرد و برای user ارزش 20 داشت و حرکتی دیگر برای CPU ارزش یک و برای user ارزش صفر داشت حرکت دوم ارجحیت دارد.

زیر برنامه mon- Rate:

این زیر برنامه با داشتن جنس مهره ارزش این مهره را پیدا می کند برای مثال ارزش مهره هایی با کد 9 تا 24 یک است و ارزش شاه برابر 40 می باشد . این زیر برنامهٔ فقط ارزیابی مهره ای انجام می دهد و دو تا مقدار ایجاد می کند RCPU و Ruser که ارزش مهره ای CPU و ارزش مهرهٔ user را نشان می دهند.

زیر برنامهٔ Time- Rate : این زیر برنامه بعد از اینکه در Move selection یک حرکت انجام شد از خانهٔ صفرام شروع به پیدا کردن مهره هایی از CPU می کند که این مهره ها توسط کاربر تهدید می شوند.

مهره هایی که تهدید می شوند یا مدافع دارند و یا بدون مدافع اند در ادامه ی بررسی ارزش مهرهٔ CPU به امتیازی که user می گیرد اضافه می شود حال اگر مهاجم جبران کرده باشد این عمل ادامه می یابد و پردازشگر شروع به پیدا کردن مدافع ها می کند و ابتدا مدافعان کم ارزشتر یعنی سرباز و سپس اسب و نیل و قلعه و در نهایت وزیر و شاه را چک می کند حال اگر مدافعی وجود داشت ، مهاجم در صورت حمله توسط مدافع گرفته خواهد شد . بنابراین ارزش مهاجم به امتیازی که CPU می گیرد اضافه می شود حال بایستی ببینیم CPU حمله کابر را جبران کرده است یا نه، اگر جبران کرده باشد به سرغ مهاجم دیگری می رویم اگر با اینکه مهاجم را گرفته است ولی نتوانسته جبران کند بنابراین حاصل این زد و خورد به نفع کاربر خواهد بود بنابراین به سراغ خانه های دیگری که مورد تهدیدند می رود و سپس محاسبه زد و خوردها و امتیازی که گرفته می شود یا از دست می رود این امتیاز گرفته شده را با امتیاز قبلی مقایسه می کند کمترین آن را به عنوان امتیاز اخذ شده از این حرکت به حساب می آورد یعنی در واقع کل کاری که این زیر برنامه انجام می دهد این است که این زیر برنامه کمترین امتیازی را که این حرکت می تواند بگیرد را پیدا می کند . این کار تا زمانی ادامه می یابد که تمامی خانه هایی که مورد هجوم کاربر هستند پیدا شده و مشخص شود که در این حرکت بیشترین امتیاز از دست داده شده چقدر است تا در زیر برنامه Moveselection با حرکتهای بعدی مقایسه شود .

زیر برنامه MAT:

این زیر برنامه در Moveselection پس از اینکه یکی از حرکت ها برای تست شدن انجام شد فراخوانی می شود این زیر برنامه مشخص می کند که اگر حرکت انجام شده بر اساس به عنوان حرکت CPU انتخاب شود User مات خواهد شد یا نه به این

صورت که با انجام شدن حرکت زیر برنامه Moveselection ، زیر برنامه کیش را فراخوانی می کند تا ببیند حرکت انجام شده باعث کیش شدن شاه شده است یا نه اگر کیش انجام شده باشد زیر برنامه Moveselection ، زیر برنامه MAT را فراخوانی می کند تا ببیند پس از کیش شدن User ، آیا User حرکتی دارد تا کیش را دفع کند پیدا کردن حرکت به این صورت است که ابتدا صفحه ی ذخیره شده سپس خانه هایی که دارای مهره ی کاربر هستند چک می شود و حرکتی که این مهره ها می توانند انجام دهند را پیدا کرده و حرکت را انجام می دهدسپس زیر برنامه کیش فراخوانی می شود تا ببیند که با انجام شدن حرکت User باز هم User کیش است یا نه اگر User کیش باشد صفحه برگردانده شده و حرکت های بعدی چک می شود و این عمل تا زمانی ادامه خواهد یافت که کیش دفع شو اگر کیش دفع نشد حرکت CPU حرکتی است که باعث کیش شدن User شده است و با انجام این کیش Userنمی تواند کیش را دفع کند بنابراین User مات است بنابراین این حرکت به عنوان بهترین حرکت CPU نمایش داده شده و Flag مات شدن User یک می شود و زیر برنامه به قسمت End Game می رود و مات شدن User را نشان می دهد.

:Userrise, CPUrise

جمع بندی :

با توجه به موارد ذکر شده بدنه اصلی برنامه به این صورت است که پس از نمایش متن خوa آمد گویی و معرفی برنامه اصلی شروع می شود این زیر برنامه در ابتدای کار با فراخوانی برنامه selection امکان انتخاب Map بازی و رنگ آن را به کاربر می دهد سپس با توجه به Map انتخاب شده ، اگر Map 1 انتخاب شده باشد به زیر برنامه Makemores رفته و به کاربر امکان تعریف حرکات جدید را می دهد و اگر Map 2 انتخاب شده باشد به زیر برنامه Mores رفته و خود حرکت های معمول در بازی شطرنج را تعریف می کند .

پس از تعریف جرکت ها لازیم است که صفحه شطرنج برای بازی آماده شود بنابراین زیر برنامه Pramery value فراخوانی شده و مهره ها را بر روی صفحه شطرنج می چیند سپس با فراخوانی زیر برنامه های Homepicture , showpicture صفحه شطرنج نشان داده می شود حال با توجه به رنگ انتخابی کاربر اگر رنگ انتخابی سفید باشد یعنی کاربر مهره های سفید را انتخاب کرده است بنابراین شروع کننده بازی کاربر خواهد بود بنابراین برنامه زیر برنامه Pointer را فراخوانی می کند تا کاربر امکان انتخاب حرکت دلخواه خود را داشته باشد و سپس نوبت عوض شده و برنامه Moveselection را فراخوانی می کند و حرکت را تجزیه تحلیل نموده و حرکت صحیح را پیدا کرده و انجام می دهد . در زیر برنامه Moveselection برنامه چک می کند که آیا حرکتی برای طرفین وجود دارد یا نه اگر طرفین حرکتی نداشتند بررسی می کند که آیا بازیکنی که حرکتی برایش موجود نیست شاه اش کیش بوده یا نه ، اگر شاه کیش باشد این بازیکن مات است وگرنع بازی پات می باشد در صورت مات بودن Flag مربوط به مات شدن بازیکن یک می شود و در صورت پات شدن بازی ، Flag مربوط به پات شدن بازی یک خواهد شد . در هر صورت اگر حرکتی برای ادامه بازی وجود نداشته باشد برنامه از حلقه Do-loop خارج شده و به EndGame Lable می رود . EndGame با توجه به Flag های مات یا پات ، مات شدن بازیکن یا پات بودن بازی را اعلام کرده سپس صفحه شطرنج را نمایش می دهد تاکاربر مات یا پات بودن را بررسی کند .

فصل چهارم

برنامه و کد نویسی پروژه :

فصل پنجم خروجی ها : سیستم طراحی شده یک سری امکانات در اختیار کاربر قرار می دهد که این امکانات عبارتند از : انتخاب رنگ مهره ، انتخاب MAP بازی و خود بازی شطرنج کاربر برای انجام همه ی کارهای خود از صفحه کلید استفاده می کند کلید های مورد استفاده برای انتخاب MAPبازی و رنگ مهره ها عبارتند از کلیدهای 8,6,5,4,2 که کلید های 2 و8 علامت "→" را بالا پایین کرده و جلوی Playmap و Colormap قرار می دهد و کلیدهای 4 و 6 ، Map و رنگ را عوض می کند برای تایید عبارت های انتخاب شده از کلید 5 استفاده می کنیم.

اگر به بازی برویم کاربر برای انتخا مهره های خود و حرکت دادن آنها از کلید های 2 و 8 برای بالا پایین آوردن محل چشمک زن استفاده می شود . و کلیدهای 4 و 6 برای حرکت دادن چشمک زن به سمت راست و چپ به کار می رود نکته ای که اینجا لازم است گفته شود اینست که برنامه به گونه ای طراحی شده که چشمک زن می تواند از یک لبه صفحه خارج شده و از طرف دیگر وارد شود . به هنگام انتخاب مهره مورد نظر رنگ زمینه آن به نشانه ی انتخاب نشدن تغییر پیدا می کند تا متوجه انتخاب شدن مهره ها شویم انتخاب مهره با کلید 5 انجام میشود و همچنین قرار دادن مهره در جای مورد نظر نیز با همین کلید انجام می گردد.

اگر کاربر Map 2را انتخاب کرده باشد به زیر برنامه تعریف حرکات جدید می رود . برای تعریف حرکت های جدید تصویر مهره ای که حرکت را برای آن تعریف می کنیم . همراه با اسم مهره نمایش داده می شود . و در پایین LCD حرکت هایی را که تعریف می کنیم نوشته می شود در بالای صفحه شماره حرکتی که داریم تعریف می کنیم نمایش داده می شود . تعریف حرکتهای جدید با استفاده از کلید های صفر تا 9 انجام می شود .

البته برنامه به گونه ای نوشته شده است که اعداد بزرگتر از 64 را قبول نمی کند حرکت های یک تا 63 نشان دهنده شماره حرکت تعریف شده است و عدد 64 نشان دهنده عدم وابستگی حرکت های بعدی به حرکتهای قبلی می باشد . وارد کردن عدد صفر نشان دهنده اتمام تعریف حرکات جدید است .

پس از وارد کردن یکی از اعداد صفر تا 64 با فشردن کلید Enter حرکت وارد شده ثبت می شود با توجه به اینکه اعداد بزرگتر از 64 قبول نیست بنابراین اولا کاربر نمی تواند حرکت های نا مفهوم تعریف کند ثانیا از این خاصیت می توان برای پاک کردن شماره وارد شده نیز استفاده کرد. به این ترتیب که مثلا اگر عدد وارد شده 10 باشد و این عدد اشتباها وارد شده باشد می توان با فشردن یکی از کلید ها عدد 10 را تبدیل به عددی بالای 100 کرد . که این عدد قابل قبول نبود . بنابراین این عدد پاک می شود و می توان عدد جدیدی را وارد کرد./

سلام بر بنده و رسول خدا، حضرت محمد (ص)، سلام بر بنده و ولیّ خدا، حضرت علی (ع)، سلام بر کاملترین مخلوق و حجّت خدا ، حضرت مهدی(عج).

Jadeye_tariki@yahoo.com

کتابخانه نیلوفر آبی: http://nilofare-abi-lib.blogfa.com

تارنگار نیلوفر آبی: http://nilofare-abi.persianblog.ir

سایت استاد محمّد رضا یحیایی: http://mry14mn.ir & http://www.mry14mn.net & http://www.mry14mn.com

شهریور ماه 1387 خورشیدی. تنظیم و ویرایش: امیر نعمتی.