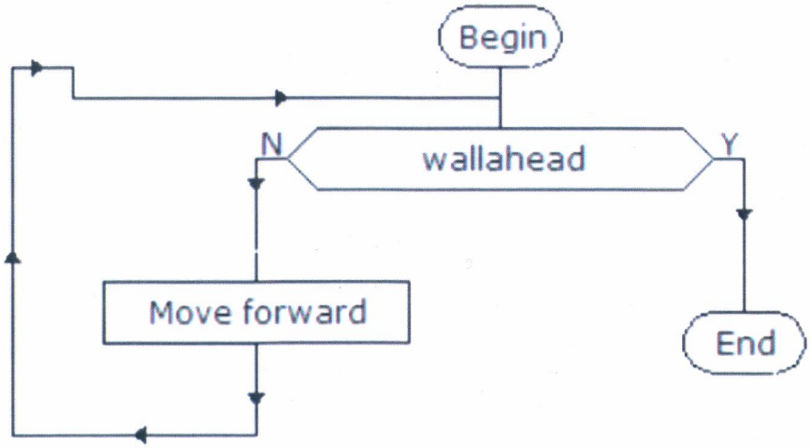
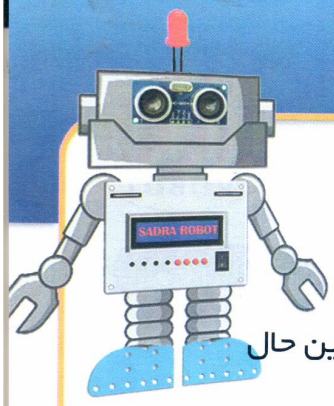
فصل چهارم

RobotProg





مقدمه

نرم افزار RobotProg طوری طراحی شده که سرگرم کننده و آموزشی بوده و درعین حال یاد گیری آن راحت باشد.

نرم افزار RobotProgابزارهای مختلفی برای شبیه سازی بر پایه برنامه نویسی بلوکی (block-based programming) دارا می‌باشد. در نرم افزار RobotProg کاربران به راحتی با طراحی فلوچارت در صفحه و اتصال بلوک‌ها به هم، مانند کنار هم قرار دادن قطعات پازل برنامه نویسی می‌کنند.

به این روش برنامه نویسی ، برنامه نویسی drag-and-drop نیز گفته می‌شود.

با توجه به تأثیر روز افزون رایانه و رشته IT در زندگی انسان، امروزه بسیاری از کشورهای پیشرفته به لزوم آموزش کامپیوتر و برنامه نویسی به دانش آموزان تأکید می‌نمایند.

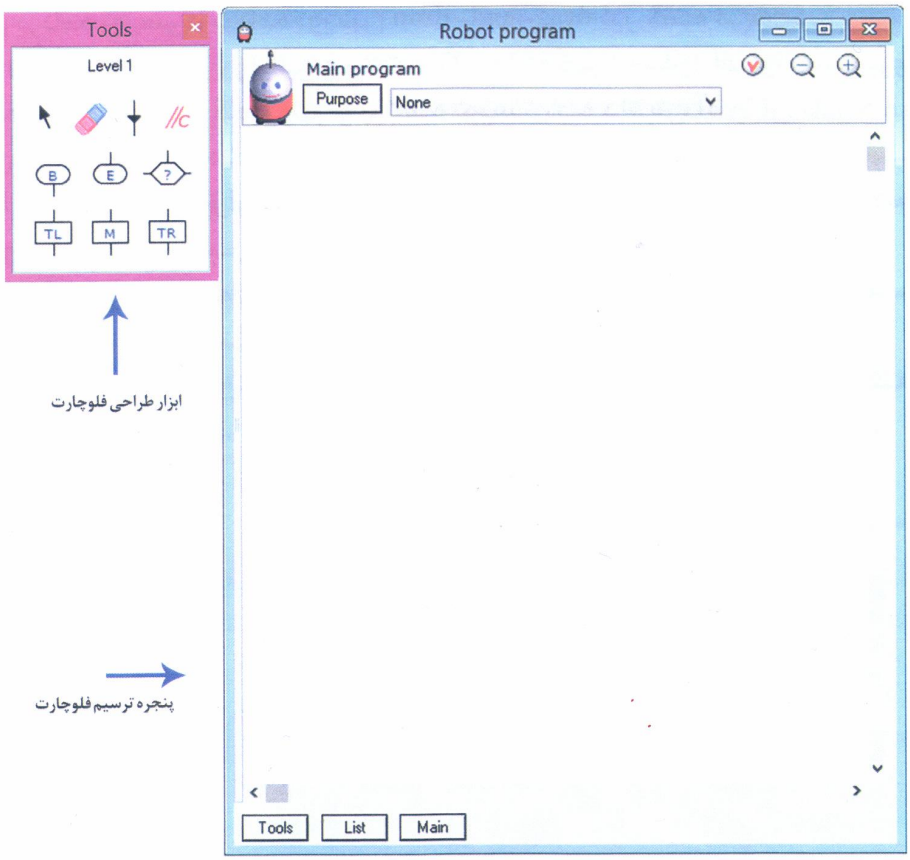
هدف اصلی از آموزش برنامه نویسی، بهبود و تقویت تفکر منطقی، تفکر محاسباتی و توانایی حل مسأله در دانش آموزان می‌باشد. تفکر محاسباتی روشی برای حل مسائل، طراحی سیستم‌ها و درک رفتار بشر بر پایه علم کامپیوتر می‌باشد.

همچنین بسیاری از شغل‌های جدید و پردرآمد در دنیای پیشرفته به خصوص در روزگار آینده بر پایه نرم افزار بنا خواهند شد.

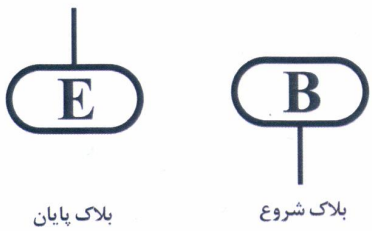
از این رو آموزش برنامه نویسی برای آماده سازی و پرورش نیروهای مستعد برای عصر دیجیتال ضروری می باشد.

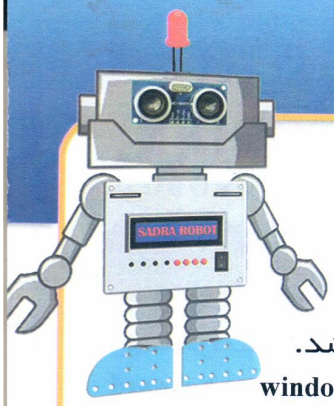
محیط برنامه

محیط این نرم افزار شامل دو پنجره می باشد :۱) پنجره ترسیم فلوچارت ۲) پنجره مربوط به ابزار طراحی فلوچارت
مطابق شکل زیر:



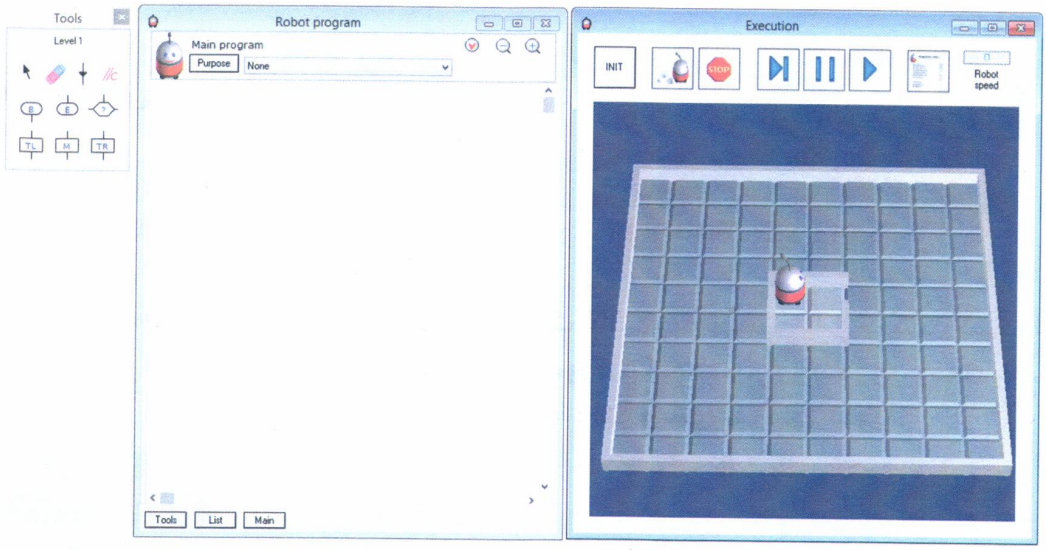
نکته: هرفلوچارت فقط می تواند دارای یک بلوک شروع باشد و می تواند چندین بلاک جهت پایان داشته باشد .





نمایش زمین اجرای نرم افزار Robotprog

این نرم افزار دارای یک زمین جهت تست و اجرای فلوچارت ترسیم شده می باشد.
جهت نمایش زمین برنامه از مسیر روبه رو استفاده نمایید: **window\Execution window**
زمین برنامه در پنجره ای دیگری باز شده و نوار ابزاری جهت کنترل اجرای برنامه بالای آن قرار دارد.



دکمه **INIT**: در صورتی که برنامه نوشته شده خطایی نداشته باشد، با کلیک بر روی این دکمه، ربات در زمین مقدار دهی شده و شما می توانید با کلیک روی ربات جهت و موقعیت آن را تعیین کنید. **Execution \ Initialization**

دکمه **Run**: جهت اجرای برنامه روی این دکمه کلیک کنید یا **Execution \ Run**

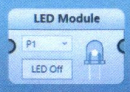
دکمه **Stop** : جهت متوقف کردن اجرای برنامه روی این دکمه کلیک کنید .

دکمه  :جهت اجرای برنامه به صورت گام به گام (مرحله به مرحله) روی این دکمه کلیک کنید .

دکمه  : جهت ایجاد وقفه در اجرای برنامه روی این دکمه کلیک کنید.

دکمه  : جهت نمایش متغیرهای تعریف شده در برنامه روی این دکمه کلیک کنید.

دکمه  : جهت تعیین سرعت ربات در حین اجرای برنامه روی این دکمه کلیک کنید.

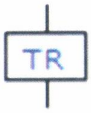


حرکت ربات

جهت حرکت ربات ۳ فرمان در برنامه وجود دارد :

Move Forward : حرکت ربات به جلو

Turn Right: چرخش ربات به راست با زاویه ۹۰ درجه



Turn Left: چرخش ربات به چپ با زاویه ۹۰ درجه

هرسه فرمان در بلاک های مستطیل شکل واقع در نوار ابزار Toolsوجود دارند .

فرمان **Move Forward** ، سبب می شود ربات یک خانه به سمت جلو حرکت کند، اما باید مواظب باشید در حرکت ربات به سمت جلو در صورتی که دیوار رو به روی ربات باشد، ربات با دیوار برخورد کرده و برنامه هنگام اجرا خطا داده و متوقف می شود .

ایجاد یک برنامه جدید

جهت ایجاد یک برنامه جدید از منوی **File** گزینه **New program** را انتخاب کنید.

نوار ابزار Tools

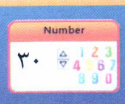
برروی این نوار، ابزارهایی جهت طراحی فلوچارت برنامه قرار دارد.

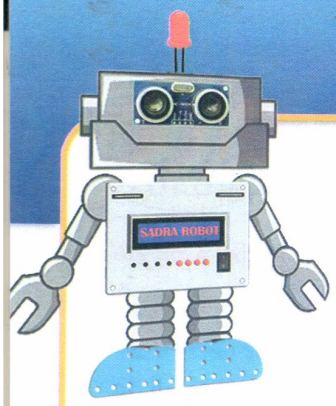
انتخاب بلاک رسم شده ←

حذف بلاک رسم شده ←

ارتباط دادن بین دو بلاک ←

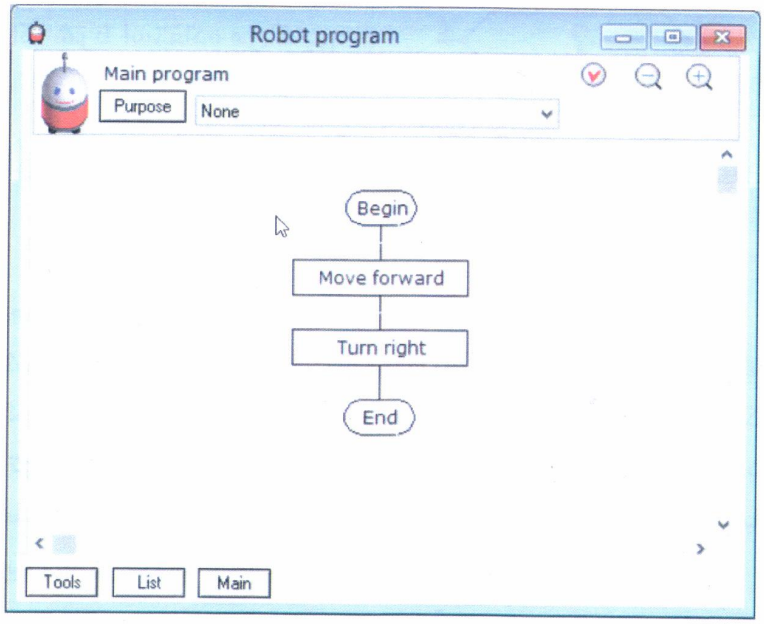
نوشتن توضیح در قسمت هایی از فلوچارت ← //c





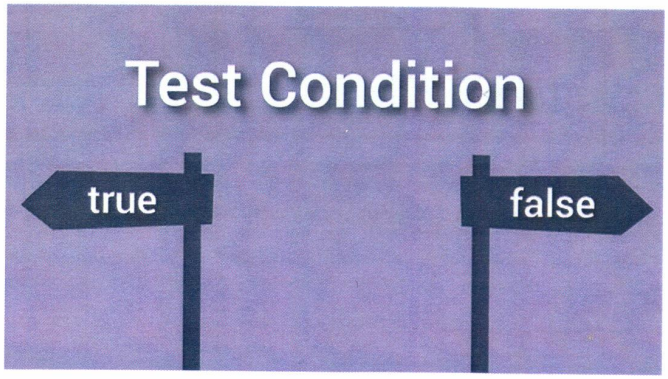
مثال ۱

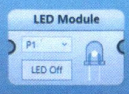
برنامه ای بنویسید که ربات یک گام به جلو برداشته و سپس ۹۰ درجه به سمت راست بچرخد.



دستورات شرطی (Conditional Statment)

دستورات شرطی یکی از پرکاربردترین موارد در دنیای برنامه نویسی به شمار می روند. در یک تعریف ساده از دستورات شرطی، می توان گفت که با آن ها امکان یا عدم امکان شرطی را بررسی می کنیم و بر اساس مثبت یا منفی بودن پاسخ، مطابق با خواست و نیازمان، کاری را انجام می دهیم؛ در دنیای واقعی نیز بارها شنیده ایم که مثلا گفته اند اگر خوب تمرین کنید، موفق می شوید یا اگر فلانی بیاید، به کوهنوردی خواهیم رفت. این نوع تعاریف در برنامه نویسی با **if** و **else** و ترکیب آن ها یعنی **elseif** استفاده می شوند.

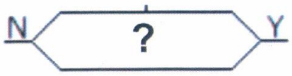




شروط منطقی

در بعضی برنامه ها حرکت ربات نیاز به رعایت شرایطی خاص در زمین برنامه دارد. به طور مثال زمانی که ربات به سمت جلو حرکت می کند به دیوار برخورد نکند .

در این مواقع از بلاک شرط استفاده می کنیم.



جهت وارد کردن شرط دلخواه داخل این بلاک، مکان نما را داخل بلاک قرار داده و دابل کلیک می کنیم .

هرشرط منطقی دو حالت دارد:

۱) در صورتی که شرط برقرار باشد، کنترل اجرای برنامه به سمت بلاک Y حرکت می کند.

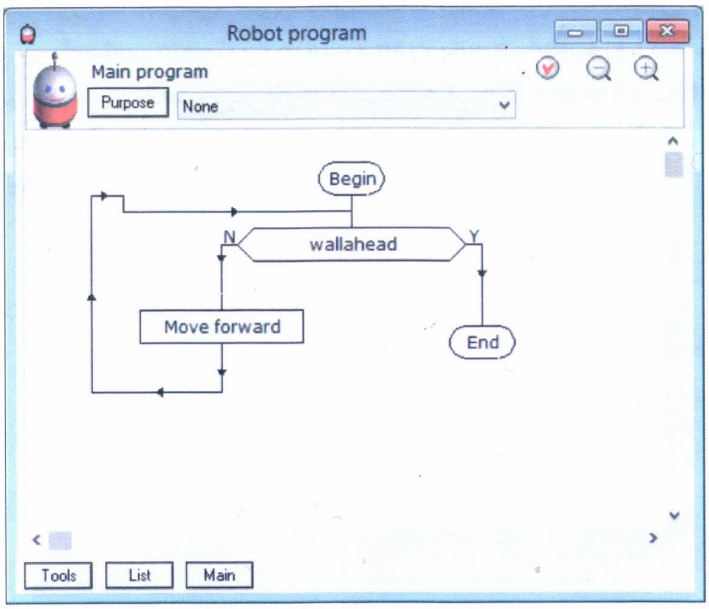
۲) در صورتی که شرط برقرار نباشد، کنترل اجرای برنامه به سمت بلاک Nحرکت می کند.

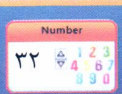
در هر حالت ما می توانیم مطابق با برنامه ای که مدنظر است یکی از راه ها را انتخاب کنیم.

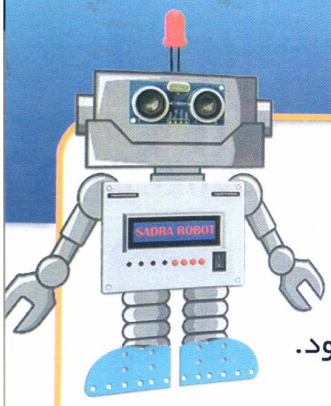
به عنوان مثال، هنگام حرکت ربات به سمت جلو اگر دیوار مقابل ربات باشد، به سمت چپ چرخیده، در غیر این صورت به سمت راست حرکت کند.

مثال ۲

برنامه ای بنویسید که ربات به سمت جلو حرکت کند و اگر به دیوار رسید، متوقف شود.







تعیین موقعیت و جهت ربات

موقعیت کاشی های زمین توسط مختصات آن به صورت (X,Y) نمایش داده می شود.

موقعیت ربات

در این نرم افزار مکان ربات توسط دو کلمه کلیدی YRobot و XRobot مشخص می شود. در طول اجرای برنامه YRobot و XRobot دو مقدار xgy را برحسب موقعیت ربات روی زمین دارند. به عنوان مثال موقعیت (۵ و۳) مشخص می کند ربات روی زمین در ردیف ۳ و ستون ۵ قرار دارد.

جهت ربات

در این نرم افزار جهت ربات توسط دو دکمه کلیدی dxRobot و dyRobot مشخص می شود.

اگر ربات رو به سمت راست زمین قرار گیرد dxRobot=۱ و dyRobot =۰

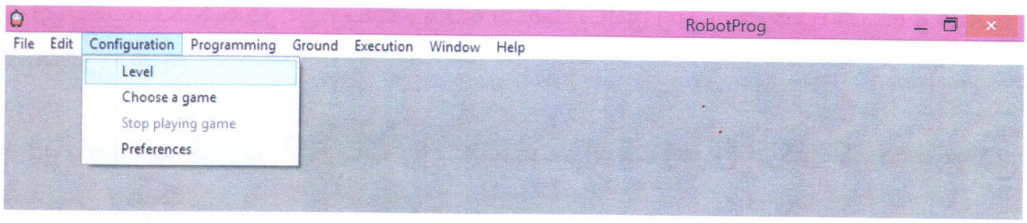
اگر ربات رو به سمت چپ زمین قرار گیرد dxRobot=-۱ و dyRobot=۰

اگر ربات رو به جلو زمین قرار گیرد dxRobot=۰ و dyRobot=۱

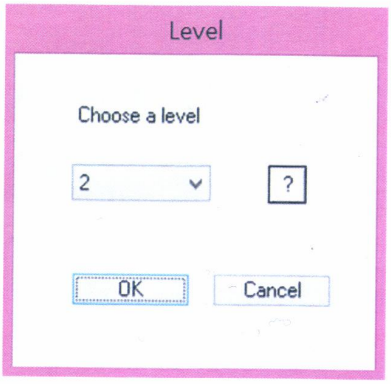
اگر ربات رو به عقب زمین قرار گیرد dxRobot=۰ و dyRobot=-۱

تنها مقادیر ممکن برای dxRobot و dyRobot مقادیر ۰، ۱ و ۱- هستند، اگر یکی از مقادیر صفر باشد، مقادیر دیگر نمی تواند صفر باشد.

موقعیت های ربات در Level۲ قرار دارند.Level انتخاب شده در جعبه ابزار Tools نمایش داده می شود.برای تغییر Level برنامه از منوی Configuration گزینهLevel را انتخاب کنید.

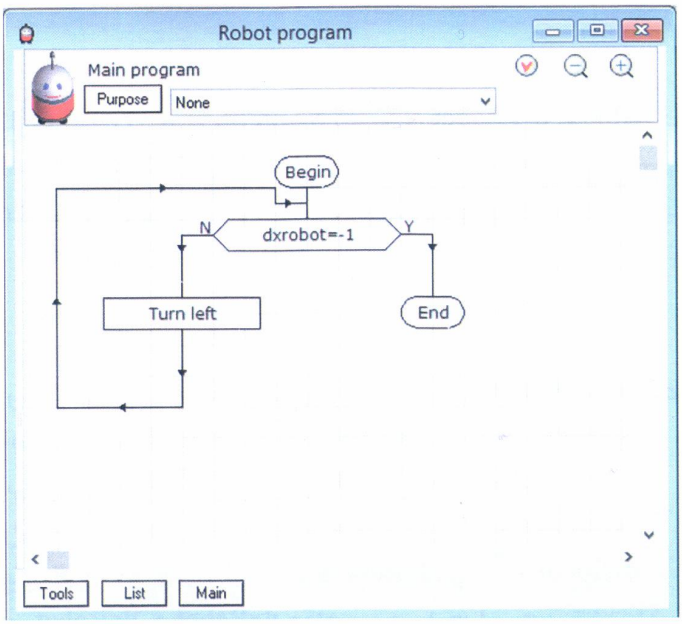


در پنجره باز شده، شماره Level مورد نظر خود را انتخاب کنید.



مثال ۳

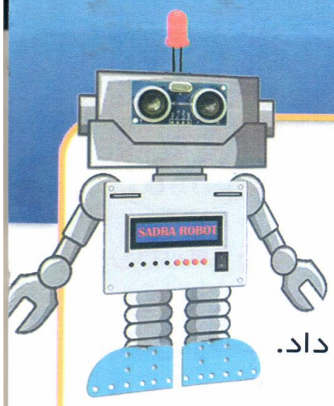
برنامه ای بنویسید که در آن ربات رو به سمت چپ زمین قرار گیرد.



تمرین

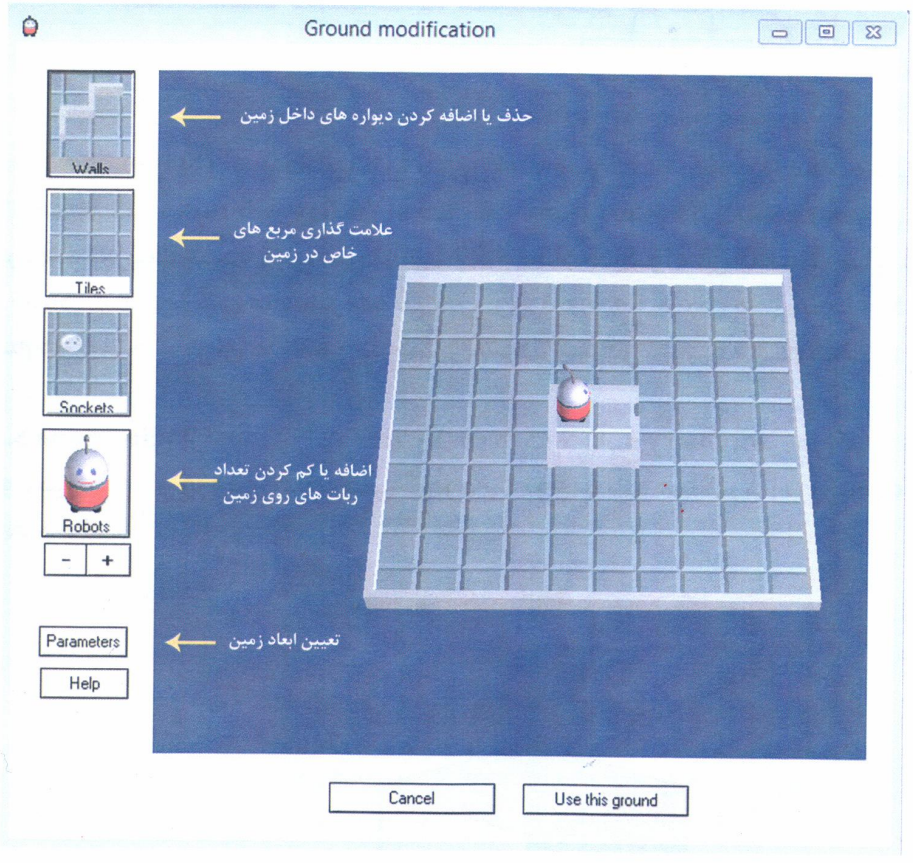
۱- برنامه ای بنویسید که در آن ربات رو به سمت بالای زمین حرکت کرده و موانع سر راه خود را تشخیص دهد.

۲- برنامه ای بنویسید که در آن ربات رو به سمت راست زمین حرکت کرده و موانع سر راه خود را تشخیص دهد.



ویرایش مشخصات زمین

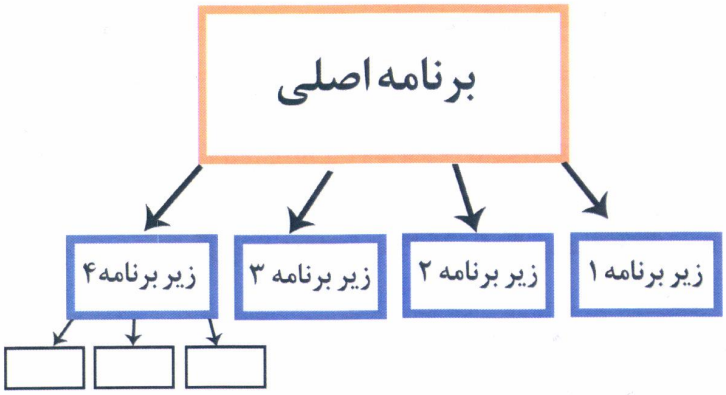
برای هر برنامه ای که نوشته می شود می توان چیدمان زمین را به دلخواه تغییر داد.
برای این کار از منوی **Ground** گزینه **Modify** را انتخاب کنید.
برای ایجاد یک زمین جدید از منوی **Ground** گزینه **new** را انتخاب کنید .
۱ – جهت حذف یا اضافه کردن دیواره های داخل زمین ابتدا روی آیکن **Walls** کلیک کرده و بعد داخل زمین کلیک کنید تا دیواره مورد نظر شما حذف یا اضافه شود.
۲ – جهت علامت گذاری مربع هایی خاص در زمین جهت تعیین محدوده زمین روی گزینه **Tiles** کلیک کنید .
۳–روی آیکن ربات کلیک کرده و روی علامت (+)زیر آن کلیک کنید، تعداد ربات ها روی زمین اضافه می شود. با کلیک روی دکمه (–) می توانید تعداد ربات های روی زمین را کم کنید .
۴–جهت تعیین ابعاد زمین روی گزینه **Parameters** کلیک کرده و طول و عرض زمین را مشخص کنید .



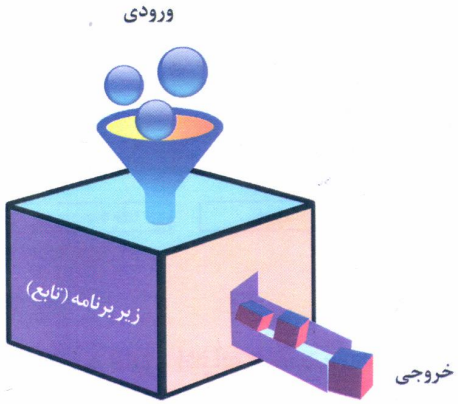
نکته:
ویرایش زمین در هنگام اجرای برنامه قابل انجام نیست.
دیواره های اطراف محدوده زمین را نمی توانید حذف کنید.

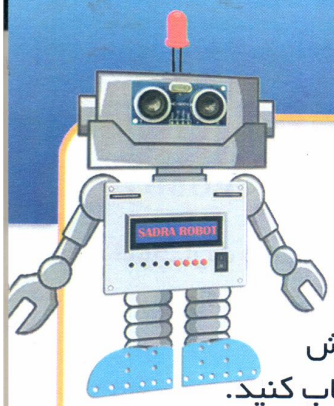
زیربرنامه (تابع) چیست ؟

گاهی اوقات برنامه ای که برای حل یک مساله نوشته می شود، بسیار پیچیده است.
در این مواقع لازم است برنامه در ابتدا تحلیل شده و به بخش های ساده تری تقسیم شود، به طوری که هر بخش یک قسمت اصلی از مساله را در برگیرد.



زیر برنامه ها (Subroutines) این قابلیت را به برنامه می دهند که برنامه به بخش های مجزا و کوچکتری تقسیم شود، به طوری که راه حل هر بخش ساده تر باشد. هر زیر برنامه فقط یک بار در برنامه تعریف شده ولی می تواند به دفعات دلخواه در قسمت های مختلف برنامه فرا خوانده شود. این ویژگی منحصر به فرد زیر برنامه ها سبب می شود از تکرار کد های شبیه به هم در قسمت های مختلف جلوگیری شده که در نتیجه حجم کمتری را نیز اشغال می کند.
هر زیربرنامه مجموعه‌ای از دستورات است که عملکرد تابع را مشخص می‌کند. هر زیربرنامه به ازای موارد که در ورودی خود می گیرد با توجه به عملکرد زیربرنامه خروجی های متفاوت خواهد داد.

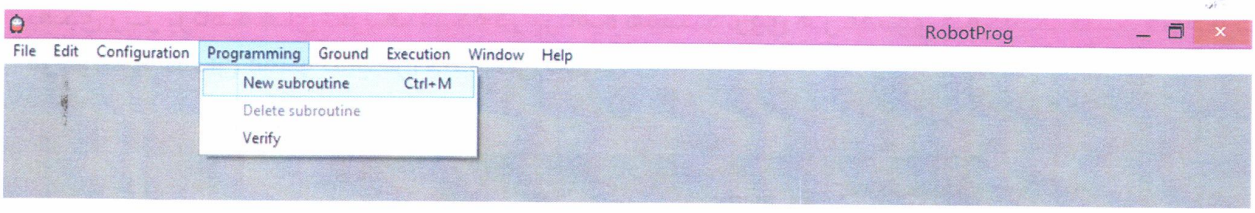




ایجاد یک زیر برنامه Subroutine

زیربرنامه ها در Level۲ قرار دارند.Level انتخاب شده در جعبه ابزار Tools نمایش داده می شود.برای تغییر Level برنامه از منوی Configurationگزینه Level را انتخاب کنید.

برای ایجاد زیر برنامه از منوی Programming گزینه new subroutine را انتخاب نمایید.

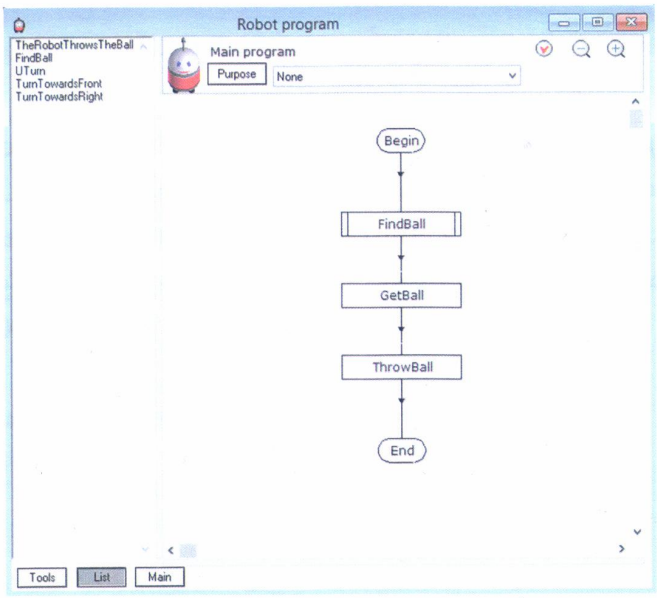


زیربرنامه جدید در پنجره Program کنار گزینه main program قرار می گیرد.
هر زیربرنامه توسط نامی که شما برای آن انتخاب می کنید، مشخص می شود. نام پیش فرض برنامه Subroutineاست.شما می توانید این نام را به دلخواه تغییر دهید.
نام زیربرنامه می تواند شامل حروف و اعداد باشد، اما نباید با عدد شروع شود، همچنین در نام زیر برنامه فاصله نباید باشد.
به عنوان مثال :
اسامی غیر مجاز: Robot Ahead ، Robotleft۴
اسامی مجاز: GoRight

فلوچارت مربوط به زیر برنامه مانند فلوچارت برنامه اصلی ساخته می شود یعنی فقط می تواند یک بلاک شروع (Begin) داشته و یک یا چند بلاک پایان (End) داشته باشد.

نمایش برنامه اصلی و زیر برنامه

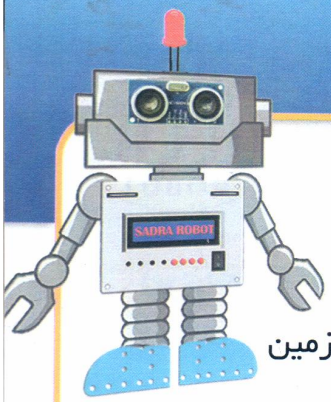
در پنجره برنامه روی دکمه List در پایین پنجره کلیک کنید تا لیست زیر برنامه ها نمایش داده شود. برای پنهان کردن لیست زیربرنامه ها نیز دوباره روی دکمه List کلیک کنید. در این پنجره اولین عضوی که نمایش داده می شود همان برنامه اصلی(main)است. عناصر بعدی زیر برنامه های تعریف شده در برنامه هستند.روی هر عضوی که کلیک کنید فلوچارت مربوط به همان عضو نمایش داده می شود .



هنگام نمایش زیربرنامه در صورتی که روی دکمهMain کلیک کنید، فلوچارت مربوط به برنامه Main نمایش داده می شود.

استفاده از زیر برنامه (فراخوانی)

هنگامی که یک زیر برنامه فراخوانی می شود، کنترل اجرای برنامه به زیر برنامه منتقل شده و فلوچارت مربوط به آن اجرا می شود. جهت فراخوانی یک زیر برنامه باید بلوک مربوط به آن رسم شود.

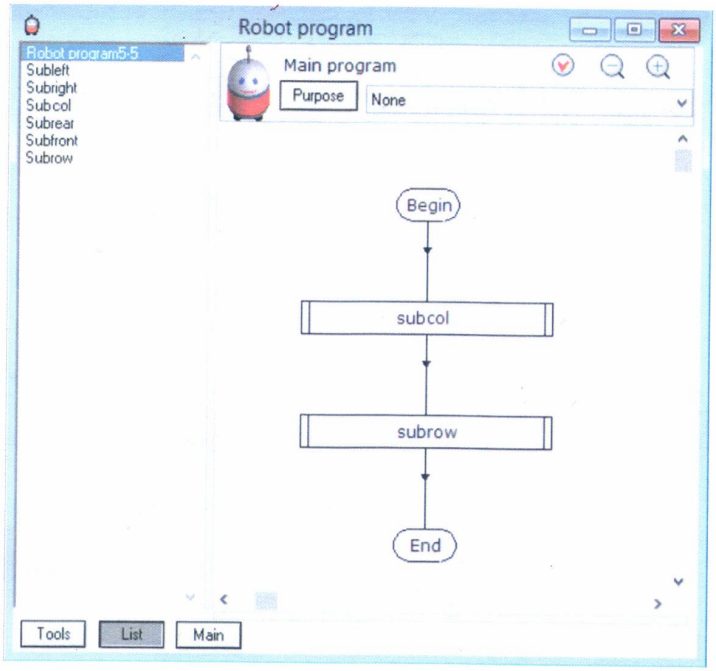


مثال ۵

برنامه ای بنویسید که ربات به خانه مرکز زمین برود.جهت حرکت ربات به مرکز زمین به دو نوع حرکت نیاز داریم:

۱-رفتن به ستون مرکزی ۲- رفتن به سطر مرکزی

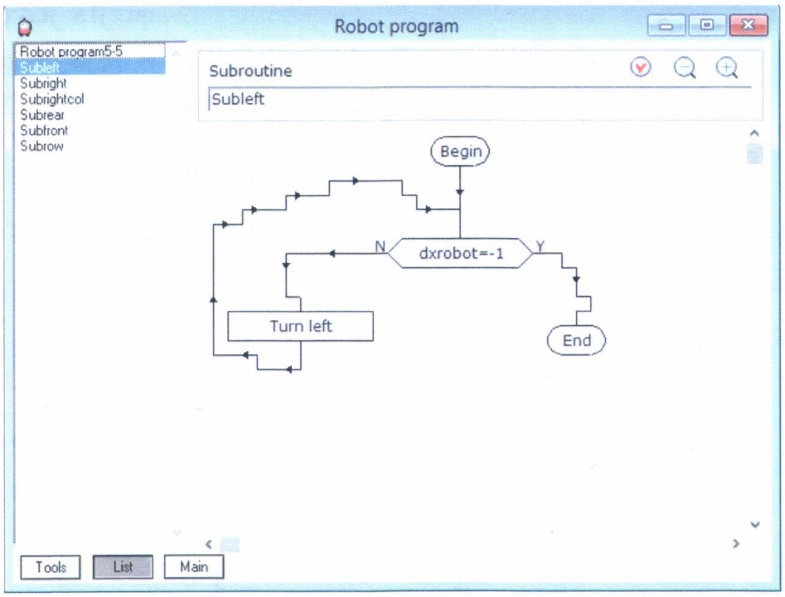
برای انجام این کار ما زیربرنامه های زیر را نیاز داریم.



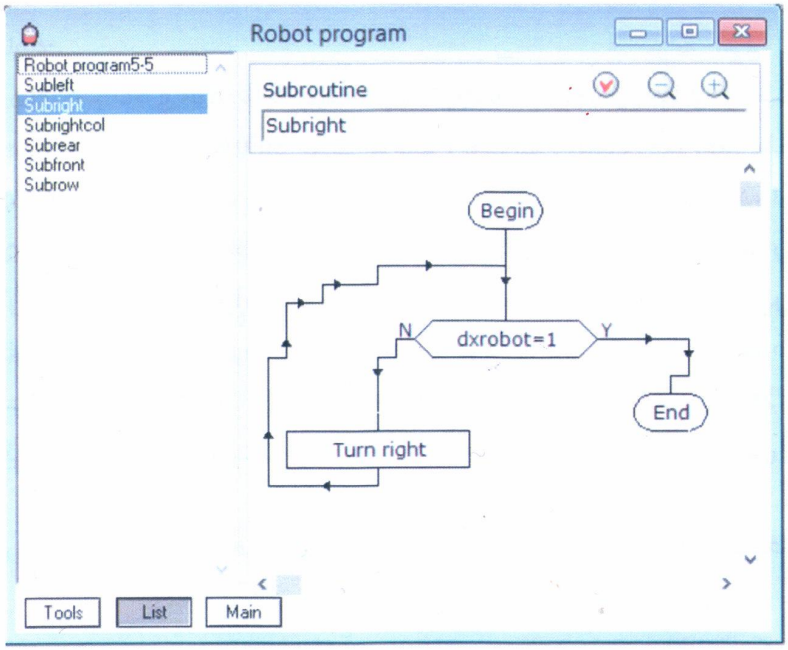
توضیحات زیر برنامه:

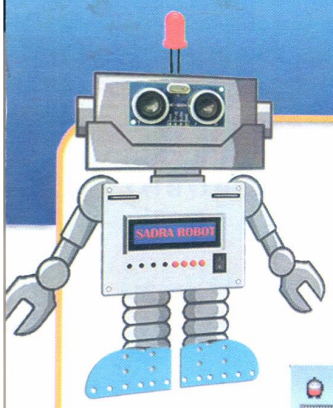
- در زیر برنامه های **Subleft** و **Subright** جهت حرکت ربات در راستای خط عمود کنترل می شود.
- در زیر برنامه های **Subfront** و **Subrear** جهت حرکت ربات در راستای خط افقی کنترل می شود.
- در زیر برنامه **Subcol** موقعیت ربات روی محور افقی چک می شود، اگر موقعیت ربات کوچکتر از عدد ۵ باشد (یعنی ربات هنوز به نقطه مرکزی نرسیده) ربات را چرخانده تا جهت ربات به سمت راست زمین قرار گیرد وگرنه ربات را به سمت چپ چرخانده تا در جهت خانه مرکزی قرار گیرد، سپس ربات را به جلو حرکت داده تا به نقطه مرکزی برسد.
- در زیر برنامه **Subrow** موقعیت ربات روی محور عمودی چک می شود، مانند زیر برنامه قبلی است با این تفاوت که ربات را به سمت جلو و عقب حرکت می دهیم تا به خانه مرکزی برسد.
- برای نمایش مکان ربات روی زمین از دو کلمه کلیدی **xrobot** و **yrobot** استفاده می شود.

زیربرنامه Subleft

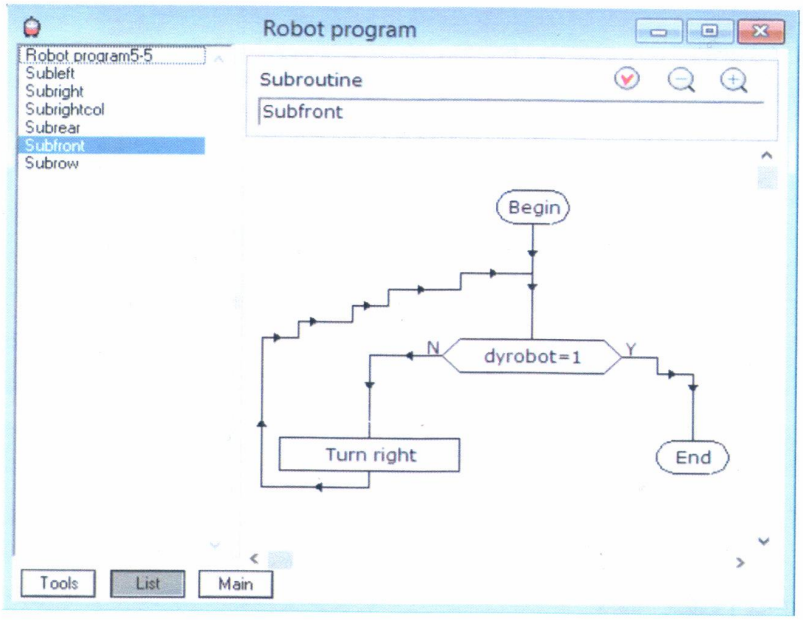


زیربرنامه Subright

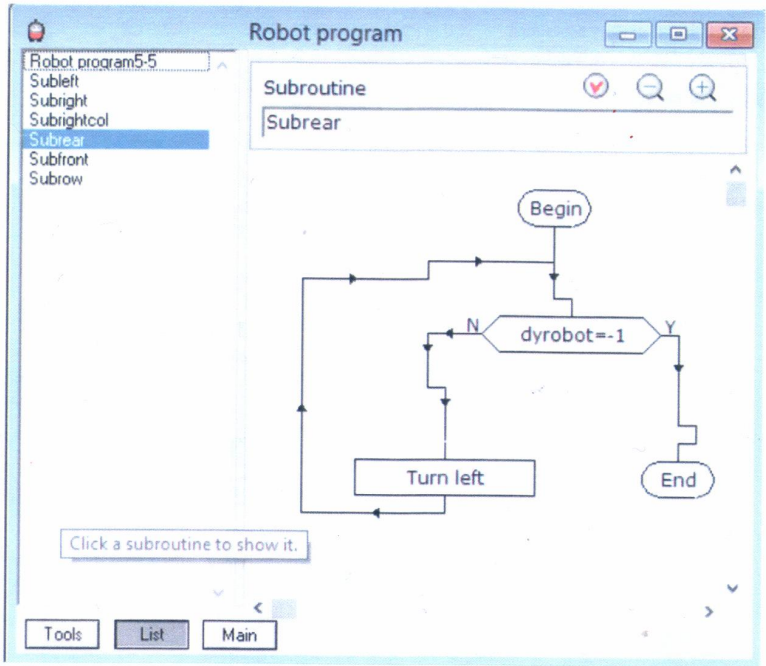




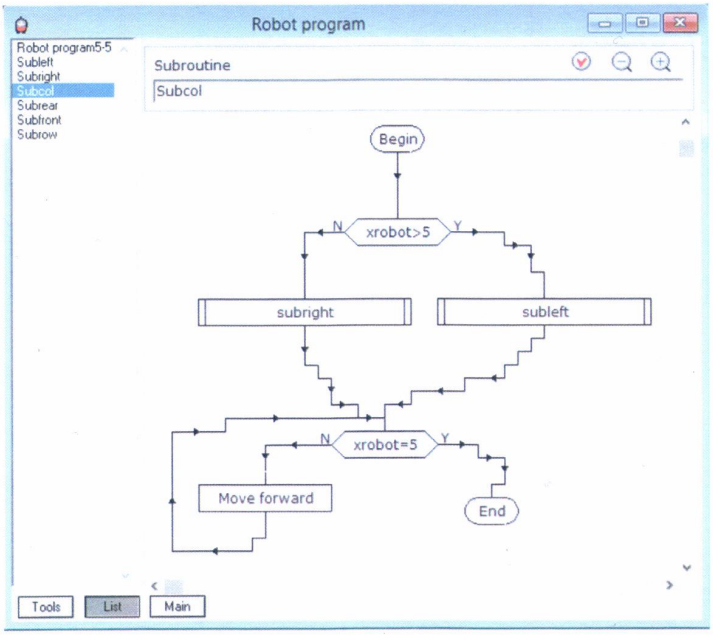
زیربرنامه Subfront



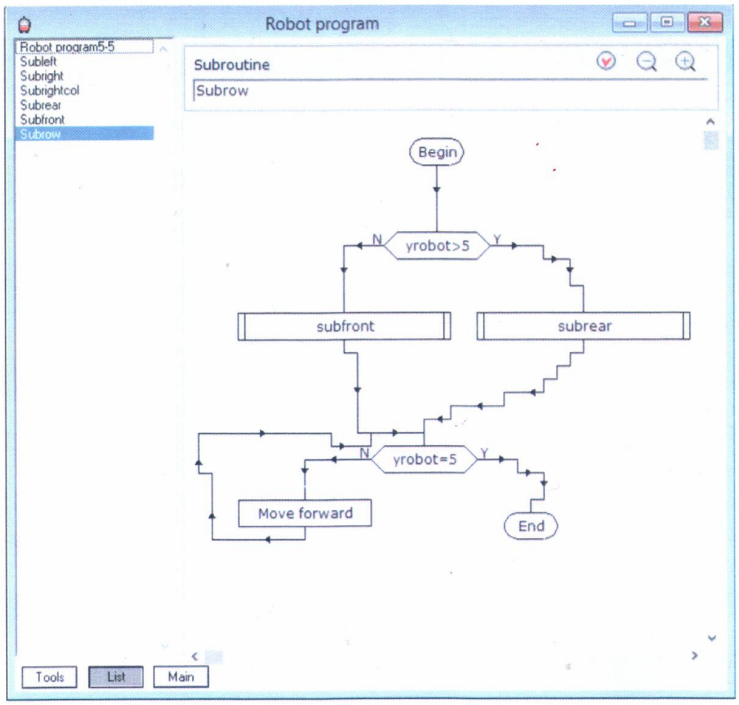
زیربرنامه Subrear

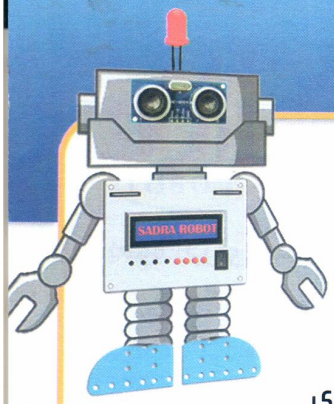


زیربرنامه Subcol



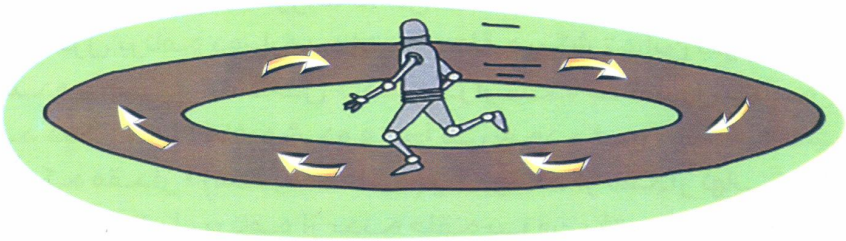
زیربرنامه Subrow



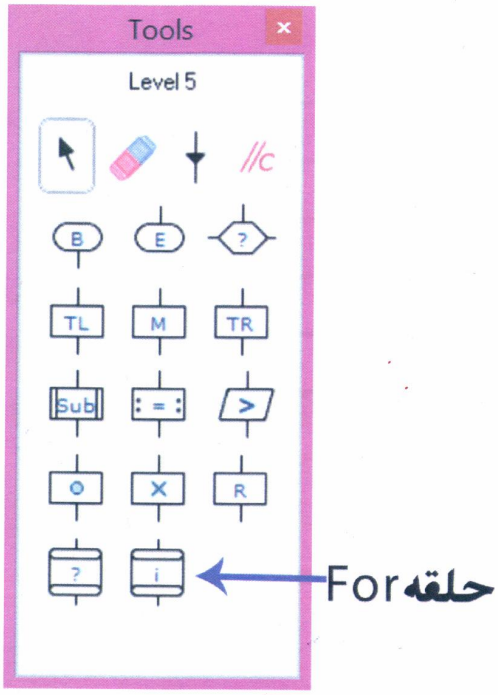


حلقه های تکرار

در مواقعی در برنامه ها لازم است که یک دستور به طور متناوب تکرار شود .
در این مواقع به جای چندین بار نوشتن یک دستور تکراری می توانیم از حلقه های تکرار استفاده کنیم.



بلوک این نوع حلقه ها در Level ۵ قرار دهد.



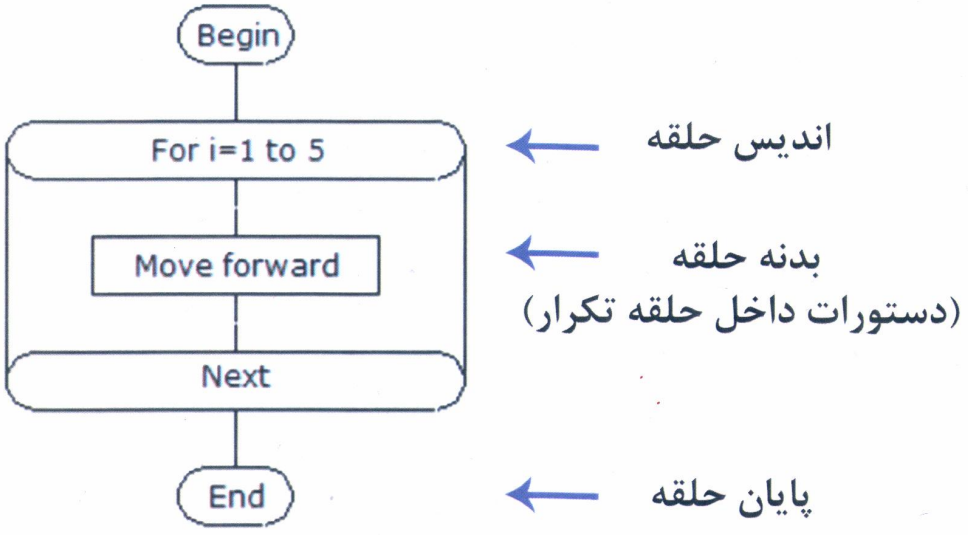
حلقه For : حلقه های تکرار پایان پذیر

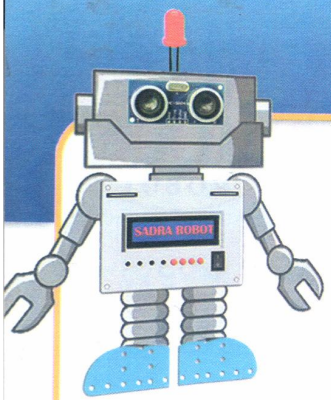
در این بلوک کلمه For و Next جزء کلمات کلیدی برنامه می باشد.
i یک متغیر است که توسط برنامه نویس تعریف می شود.

نحوه اجرای حلقه های For به صورت زیر است:

For i = ۱ to ۴

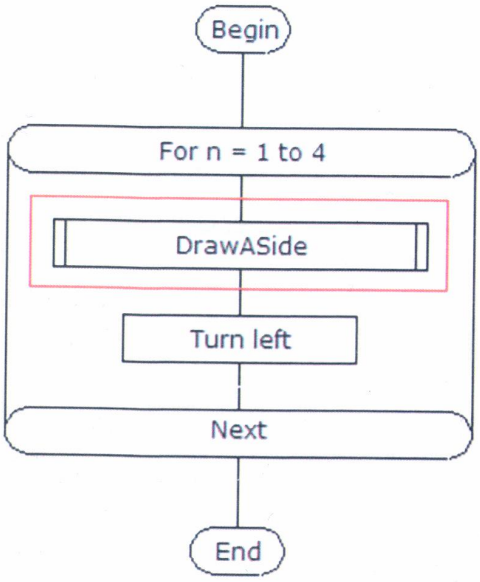
درابتدا مقدار i (شمارنده یا اندیس حلقه) برابر با مقدار اولیه یعنی ۱ می شود و کنترل برنامه وارد بلوک حلقه شده و دستورات داخل حلقه اجرا می شوند، وقتی کنترل حلقه به کلمه Nextرسید، کنترل برنامه دوباره به ابتدای حلقه برگشته،این بار یک واحد به i اضافه می شود، i=۲ شده و دستورات داخل حلقه تکرار شده تا کنترل برنامه به Next برسد این بار یک واحد دیگر به i اضافه شده و i=۳ می شود و این روند به همین شکل ادامه می یابد تا وقتی که مقدار i (شمارنده حلقه) یک واحد از مقدار نهایی بیشتر شود یعنی i=۵ که در این صورت کنترل برنامه از بدنه حلقه بیرون رفته و دستورات بعد از آن اجرا می شود.

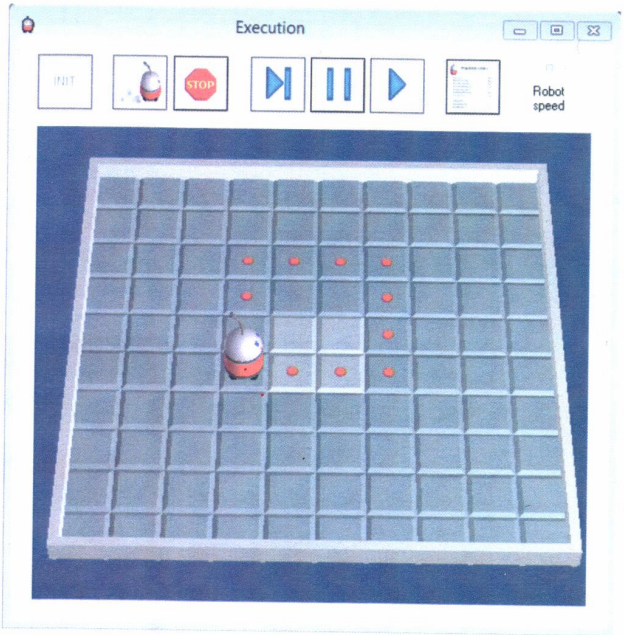




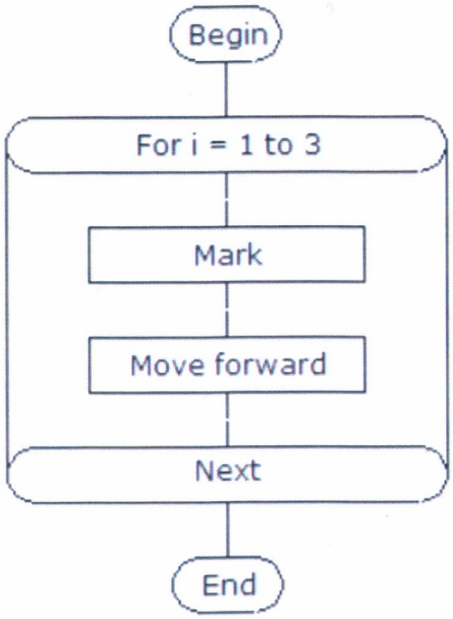
مثال ۶

برنامه ای بنویسید که ربات در آن یک مربع ترسیم کند.
جهت رسم یک نشانه، ربات از دستور **Mark** استفاده می کند.
در این برنامه ربات یک ۴ ضلعی ترسیم کرده که هر ضلع آن با چهار دایره مشخص می شود. جهت نوشتن این برنامه ابتدا یک زیر برنامه با نام **Drawside**، جهت رسم دایره های یک ضلع ایجاد می کنیم. چون هر ضلع با چهار علامت دایره رسم می شود بنابراین به حلقه **For** نیاز داریم.
با این زیر برنامه یکی از اضلاع ترسیم می شود. حال فلوچارت برنامه اصلی را به صورت زیر طراحی می کنیم .
در برنامه اصلی ربات یک ضلع را ترسیم کرده، به سمت راست پیچیده و ضلع بعد را رسم می کند.

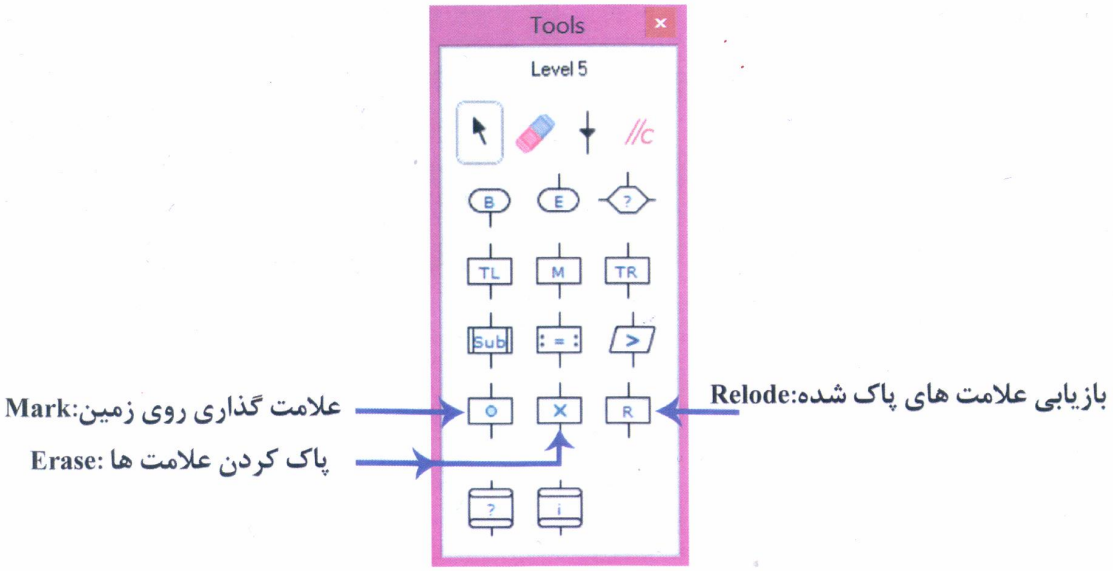


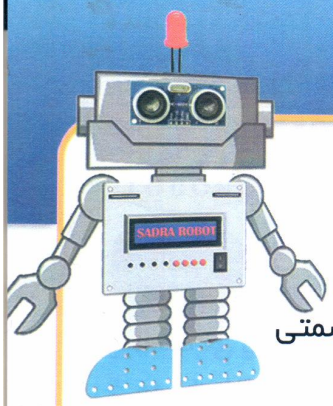


زیربرنامه DrawASide



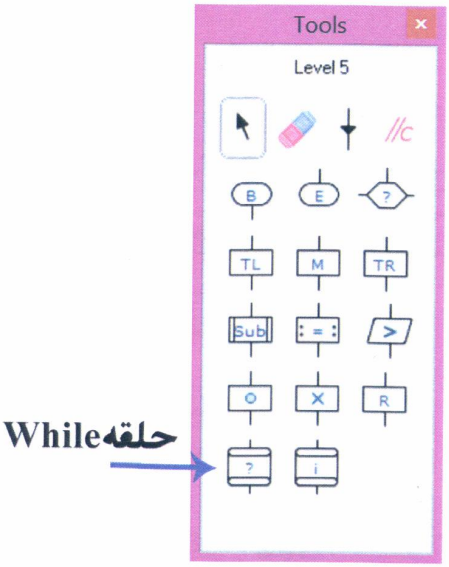
دستورات مربوط به علامت گذاری روی زمین در Level۵ به بعد قرار دارند. در شکل زیر این دستورات را می بینید.





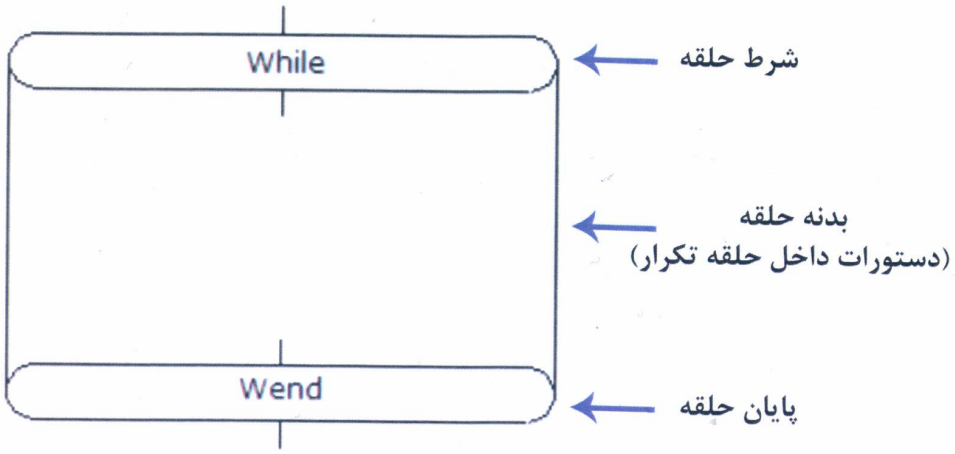
حلقه While (حلقه تکرار پایان ناپذیر)

ساختار تکرار (حلقه تکرار) به برنامه نویس این امکان را می دهد که برنامه، قسمتی از دستورات را تا هنگامی که شرط خاصی برقرار است، تکرار کند.



دستور while

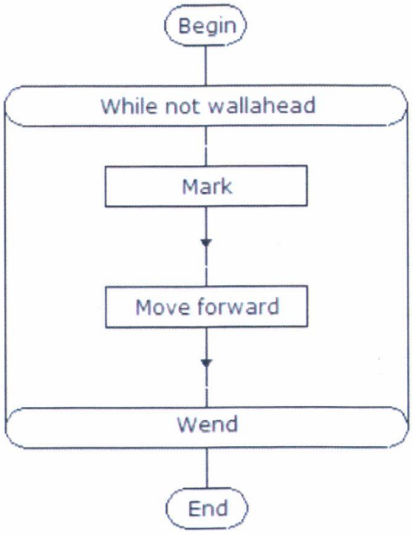
در این دستور ابتدا شرط موجود در جلوی **while** بررسی می شود، اگر شرط دارای ارزش درستی باشد دستورات داخلی ساختار فایل اجرا می شود و دوباره کنترل اجرای برنامه به ابتدای حلقه برمی گردد و این روند ادامه پیدا می کند. اگر شرط نادرست باشد کنترل اجرای برنامه از حلقه خارج می شود. دقت کنید که شرط حلقه باید در بدنه ی برنامه نصب گردد تا حلقه خاتمه پیدا کند وگرنه حلقه ی بی نهایت ایجاد می شود.



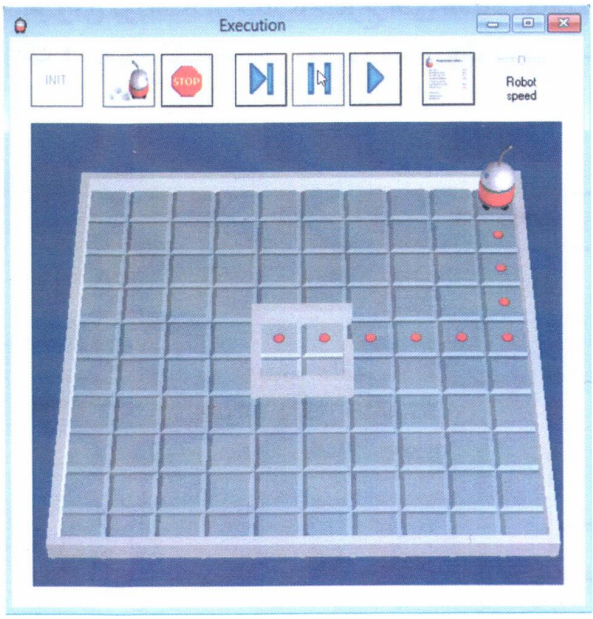
مثال ۷

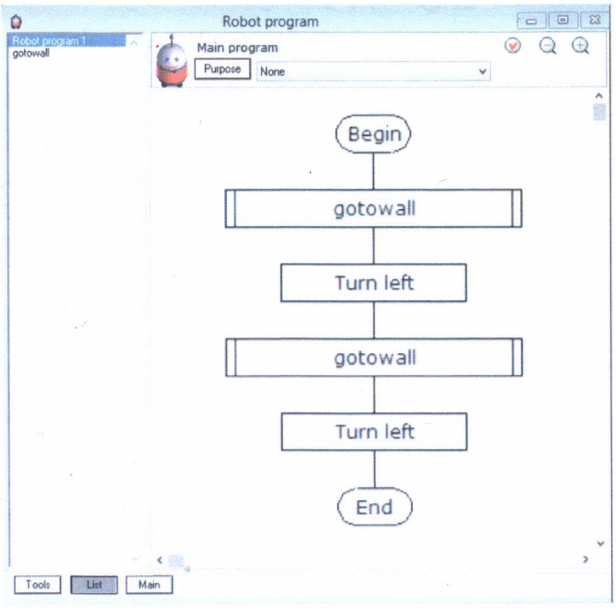
برنامه ای بنویسید که ربات در آن به سمت گوشه بالای زمین حرکت کرده و مسیر حرکت خود را علامت گذاری کند.

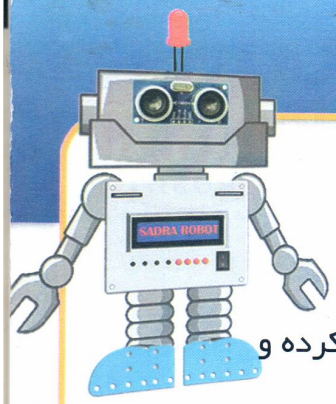
زیربرنامه gotowall



برنامه اصلی

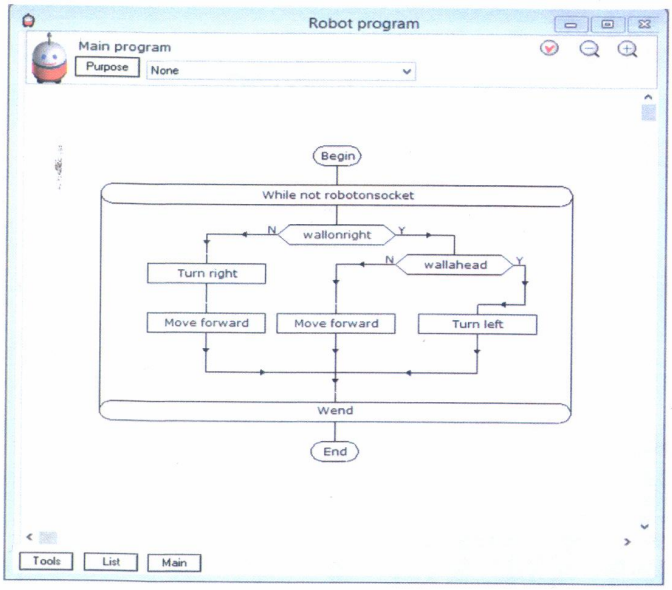


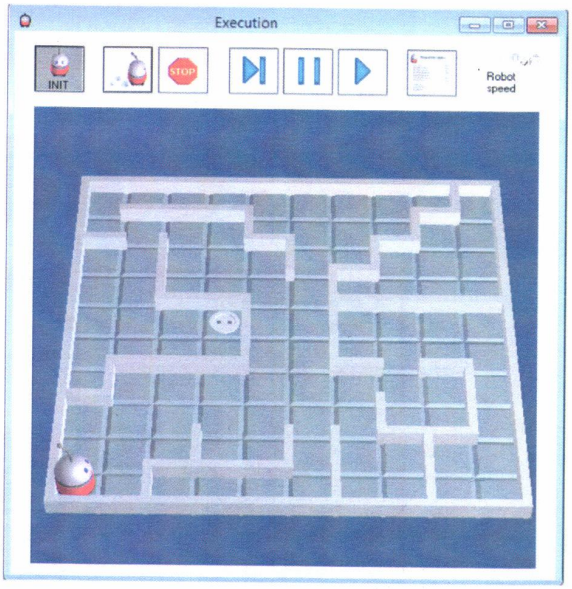




مثال ۸

برنامه ای بنویسید که ربات در آن روی زمین ماز مانند، مطابق شکل زیر حرکت کرده و حفره را پیدا کند.





تمرین

۱ – برنامه ای بنویسید که ربات یکی از حروف الفبای انگلیسی را روی زمین رسم کند.
۲– برنامه ای بنویسید که ربات اسم خودتان را روی زمین رسم کند.
۳– برنامه ای بنویسید که ربات به خانه ای با مختصات(۶، ۴) برود.
۴– زمین دلخواهی را طراحی کنید و برنامه ای بنویسید که ربات روی مسیر طراحی شده حرکت کرده و مسیر را نشانه گذاری کند.