تا این جا با مفاهیم اولیه متغیر آشنا شده‌ و سه نوع متغیر int، float و bool را شناخته‌اید. در این بخش می‌خواهیم با تعریف دقیق‌تر متغیر، بحث‌های فنی حول ذخیره‌سازی متغیرها، استفاده‌ی آن‌ها و انواع آن با این مقوله‌ی مهم برنامه‌نویسی بیشتر آشنا شویم.

متغیر روشی برای نامگذاری و ذخیره‌ی مقداری است که بعدا در برنامه استفاده می‌شود مثل داده‌ی یک حسگر.

سینتکس کلی تعریف متغیرها و مقداردهی آن‌ها به صورت زیر است:



که DataType نوع متغیر، name نام متغیر و value مقدار متغیر است.

تجربه ثابت کرده‌است بهتر است در برنامه‌نویسی آردوینو ابتدا متغیر تعریف شود و سپس مقداردهی شود یعنی به‌ صورت زیر:



این روش از لحاظ نظری فرقی با روش اول ندارد، اما در بعضی مواقع دیده شده است که روش اول باعث بروز مشکلاتی می‌شود.

بیت و بایت

بیت کوچک‌ترین واحد ذخیره‌سازی کامپیوتر است که می‌تواند صفر یا یک باشد. هشت بیت کنار هم یک بایت را تشکیل می‌دهند که واحد رایج برای مشخص کردن مقدار حافظه است. از آن جا که هر بایت 8 بیت دارد و هر بیت 2 حالت، پس می‌توان برای هر بایت 2 به توان 8 یعنی 256 حالت مختلف در نظر گرفت.

هر نوع متغیر تعداد مشخصی بایت را به خود اختصاص می‌دهد که این تعداد نشان‌دهنده‌ی ظرفیت آن متغیر است. مثلا متغیر int که جلوتر با آن آشنا خواهید شد، 2 بایت را اشغال می‌کند پس می‌تواند 2 به توان 16 مقدار مختلف را در خود ذخیره کند.

bool (boolean)

bool ساده‌ترین متغیر برنامه‌نویسی است. این متغیر نماینده و ذخیره‌ساز داده‌های منطقی است؛ داده‌ای که می‌تواند فقط صفر یا یک باشد. کد زیر نحوه‌ی مقداردهی این متغیر را نشان می‌دهد.



دقت کنید این متغیر غیر از موارد ذکر شده مقدار دیگری نمی‌پذیرد. البته اگر هر مقداری غیر از صفر، مثلا "ali"، به آن نسبت داده شود، آن را یک درنظر می‌گیرد.

boolean نیز کاملا مانند bool است با این تفاوت که مختص برنامه‌نویسی آردوینو است. بنابراین توصیه می‌شود به علت استاندارد نبودن آن در اصل زبان C++ از آن استفاده نشود.

int

متغیر int برای ذخیره‌سازی اعداد صحیح به‌کار می‌رود.



باید توجه کرد بازه‌ی پوشش آن محدود است زیرا بسته به نوع میکروکنترلر، آردوینو فضای ذخیره‌سازی خاصی را به آن اختصاص می‌دهد. در این جدول فضای ذخیره‌سازی هر برد آردوینو و بازه‌ی پوشش آن مشخص شده‌است:

|  |  |  |  |
| --- | --- | --- | --- |
| خانواده‌ی میکروکنترلر | مقدار حافظه‌ی اشغالی(بیت) | بازه | برد نمونه |
| ATmega | 16 | [-32768,32767] | UNO |
| SAMD | 32 | [-2147483648,2147483647] | Zero |

اگر مقداری فراتر از بازه به ‌آن نسبت داده شود، رفتار میکروکنترلر در ذخیره‌سازی آن پیش‌بینی‌پذیر نیست. یک احتمال آن است که از بیشینه مقدار خود فراتر رود و دوباره به ابتدای بازه برگردد، اما همیشه اینگونه نیست.

float

این متغیر برای ذخیره‌سازی اعداد اعشاری مثبت و منفی استفاده می‌شود.



آردوینو به آن 32 بیت (4 بایت) حافظه از مموری را اختصاص می‌دهد. بازه‌ی مقداردهی این متغیر به این صورت است: [ -3.4028235E+38,3.4028235E+38]

float تنها می‌تواند تا 6 یا 7 رقم را در خود ذخیره کند که این تعداد شامل تمام رقم‌های عدد می‌شود نه فقط رقم‌های بعد از اعشار.

متغیر float در محاسبات دقیق نیست و همین طور زمان زیادی را در محاسبات از میکروکنترلر می‌گیرد. بنابراین در کارهای دقیق و یا با سرعت بالا استفاده از آن توصیه نمی‌شود.

double

در آردینو double و float کاملا یکسانند. برخلاف بقیه‌ی پلتفرم‌ها که می‌توان با استفاده از متغیر double به دقت بیشتری رسید، در آردوینو این دو تفاوتی ندارند.

long

long مانند int اعداد صحیح را ذخیره می‌کند با این تفاوت که مقدار بیشتری از حافظه را اشغال می‌کند که باعث می‌شود بازه‌ی بزرگتری از اعداد را پوشش دهد. این متغیر 4 بایت را اشغال می‌کند که بازه‌ی [-2147483648,2147483647] را می‌پوشاند. اگر بخواهیم عددی که در بازه‌ی int قرار می‌گیرد را به صورت متغیر long ذخیره کنیم، باید انتهای آن مقدار حرف L را قرار دهیم. مثال زیر نحوه‌ی مقداردهی این متغیر را نشان می‌دهد:



byte

byte متغیری 8 بیتی (1 بایتی) است که اعداد صحیح بی‌علامت بین 0 تا 255 را ذخیره می‌کند.

char

char برای ذخیره‌سازی کاراکترها به‌کار می‌رود. این متغیر 1 بایت از حافظه را می‌گیرد و فقط یک کاراکتر را در آن ذخیره می‌کند. کد زیر نحوه‌ی تعریف این متغیر را نشان می‌دهد:



char کاراکترها را بر اساس کد ASCII ذخیره می‌کند. این شیوه‌ی کدگذاری هر کاراکتر را با یک عدد از 0 تا 127 مرتبط می‌کند. جدول کامل کدهای ASCII را اینجا ببینید. به همین علت، می‌توان به صورت زیر نیز به char مقداردهی کرد:



دقت کنید که عدد 65 کد کاراکتر A است.

array

array گروهی از متغیرهاست که با نمایه‌ی عددی قابل دسترسی‌اند. در حقیقت array راهی برای ذخیره‌سازی و دسترسی به تعدادی از داده‌هاست که می‌خواهیم آن داده‌ها را یکجا ذخیره کنیم. سینتکس واحدی برای ساخت array وجود ندارد اما مثال‌های زیر همگی راهی برای تعریف array هستند:



خط اول تنها متغیر name1 را با 6 نمایه می‌سازد بدون اینکه مقداردهی کند و مشخص می‌کند مقادیر آن از نوع int اند.

خط دوم متغیر name2 را می‌سازد و با مقداردهی تعداد نمایه‌های آن بدون آنکه ذکر شود مشخص می‌شود و مشخص می‌کند مقادیر آن از نوع int اند.

خط سوم متغیر name3 را می‌سازد که هم تعداد نمایه‌های آن مشخص شده و هم مقداردهی شده است و مشخص می‌کند مقادیر آن از نوع int اند.

خط چهارم متغیر name4 مانند خط سوم است با این تفاوت که مقادیر از نوع float تعیین شده‌اند.

و در نهایت خط پنجم متغیر name5 را که از 6 نمایه تشکیل شده است می‌سازد و مشخص می‌کند این متغیر از نوع char است که به همین خاطر مقداردهی آن با علامت " " انجام شده‌ است. دقت کنید تعداد نمایه‌ یکی از تعداد حروف بیشتر است که آن برای ذخیره‌سازی کاراکتر null است.

فراخوانی یک نمایه از array به صورت زیر صورت می‌گیرد:



توجه: نمایه‌ها از صفر شروع می‌شوند یعنی اولین مقدار با نمایه‌ی 0 علامت‌گذاری می‌شود نه 1.

مقداردهی به هر نمایه نیز به صورت زیر صورت می‌گیرد:



String

String آرایه‌ای از کاراکترهاست. سینتکس آن به صورت زیر است:



به تفاوت و شباهت‌ بین این نحوه‌ی ذخیره‌ی hello و استفاده از آرایه‌ی کاراکتری در بخش array دقت کنید. در آن جا برای ذخیره‌ی این کلمه آرایه‌ای 6 خانه‌ای از نوع char درست کردیم. اما در این جا، مستقیما به متغیرمان مقدار "hello" را دادیم. در حقیقت متغیر String همان کار تعریف آرایه‌ی از نوع char را برای ما انجام می‌دهد.

توجه: مقدار متغیر String حتما باید داخل " " (دابل کوتیشن) قرار بگیرد نه ' ' (سینگل کوتیشن). سینگل کوتیشن برای مقداردهی یک char کاربرد دارد.

توجه: به بزرگ بودن S در نام این متغیر توجه کنید.

توجه: string و String تفاوت دارند. به آن متغیری که از راه روش تعریف آرایه از نوع char درست شود string می‌گویند و به متغیری که از روش دوم درست شود String.

قبل از آنکه به انواع متغیرها بپردازیم، لازم است با مفهوم scope آشنا شویم.

scope

scope محلی است که متغیر در آن شناخته می‌شود. بسته به آنکه متغیر در کجای برنامه تعریف شود، مکان شناخت آن نیز متفاوت می‌شود. در آردوینو در 3 مکان می‌توان متغیر را تعریف کرد:

1. داخل setup() ، loop() و هر تابع تعریف شده‌:

هر متغیری که داخل این دو تابع اصلی آردوینو یا داخل هر تابع تعریفی خودمان، تعریف شود، فقط در داخل آن تابع شناخته و به کار برده می‌شود. این متغیرها بعد از به اتمام رسیدن کار تابع نابود می‌شوند و با دوباره فراخوانده‌شدنشان دوباره ساخته می‌شوند. به این متغیرها local می‌گویند.

1. داخل ساختارهای حلقه‌ای مانند for() و while():

برای کنترل این حلقه‌ها متغیرهایی تعریف می‌شوند که فقط داخل این حلقه‌ها کاربرد دارند. مانند حالت 1، این متغیرها با فراخواندن حلقه به وجود می‌آیند و با اتمام کارشان از بین می‌روند.

1. خارج از موارد ذکرشده:

متغیرهای این دسته متغیرهای global نامیده می‌شوند و در تمام برنامه شناخته می‌شوند و نابود نمی‌شوند.

مثال زیر تمام موارد بالا را نشان می‌دهد:



مطالبی که در ادامه می‌آید انواع جدید متغیر نیست، بلکه عملگرهایی‌اند که بر روی متغیرها تاثیراتی می‌گذارند.

unsigned

نوشتن unsigned قبل از int و long این متغیرها را به متغیرهایی تبدیل می‌کند که فقط عدد بی علامت (مثبت و صفر) می‌پذیرند. از آن جا که مقدار حافظه‌ی اختصاص داده به این متغیرها تغییر نمی‌کند، پس بازه‌ی مقدارگیری این متغیرها در بخش مثبت اعداد دوبرابر می‌شود. به مثال‌های زیر دقت کنید:



const

گاهی لازم است متغیری تعریف کنیم که مقدارش به هیچ وجه تغییر نکند. در این صورت کافی است const را قبل از نوع متغیرمان بگذاریم:



دقت کنید که متغیری که با const تعریف شده است فقط یک بار می‌تواند مقداردهی شود.

static

با نوشتن static قبل از نوع متغیر، اگر آن متغیر local باشد از نابودی آن جلوگیری می‌کنیم. در مبحث scope گفتیم که متغیری که در یک تابع تعریف شود، بعد از اتمام کار تابع از بین می‌رود که این بدان معناست مقدار ذخیره شده نیز دیگر در دسترس نخواهد بود. اما گاهی لازم است این مقدار و متغیر را نگه داریم. کد زیر نحوه‌ی این کار را نشان می‌دهد:

