

MODUL ROBOTIC SCHOOL
BIDANG SOFTWARE
PENGENALAN BAHASA PEMROGRAMAN DAN IDE

A. BAHASA PEMROGRAMAN

Bahasa pemrograman adalah bahasa komputer yang digunakan programmer untuk mengembangkan program software, script, atau sekumpulan set instruksi agar bisa dieksekusi oleh komputer. Dan pemrograman dalam arti luasnya adalah kegiatan menganalisis kebutuhan, perencanaan tahapan/planning, perancangan dan perwujudan atau implementasi program atau code.

sederhananya, pemrograman adalah memberikan set instruksi ke komputer untuk dieksekusi atau dijalankan, dan bahasa pemrograman adalah alat yang digunakan untuk menulis instruksinya. Jenis bahasa pemrograman ada banyak namun yang paling banyak digunakan adalah bahasa pemrograman Python, C/C++, Java, PHP, dan JavaScript. Dalam robotik untuk memprogram suatu microcontroller biasanya bahasa yang sering digunakan adalah bahasa C/C++ dan Python.

1. Struktur Bahasa Pemrograman Arduino dan ESP

Suatu program Arduino/ESP umumnya terdiri atas instruksi void setup () dan void loop (). Instruksi void setup() digunakan untuk menginisialisasi variabel-variabel yang akan digunakan, dan hanya dijalankan satu kali saat Arduino mulai menyala. Sedangkan instruksi void loop() digunakan untuk menjalankan suatu siklus program, yang akan dilakukan terus-menerus hingga Arduino mati/reset. Berikut ini adalah beberapa fungsi dasar pada Arduino.

1) pinMode(pin, SET)

Fungsi ini digunakan untuk menginisialisasi sebuah pin, dan menentukan pin tersebut akan digunakan sebagai input maupun output. Nilai SET dapat diisi OUTPUT atau INPUT, tergantung dari kebutuhan. Sedangkan nilai pin adalah nomor pin pada mikrokontroler yang akan diset sebagai input atau output. Contoh: **pinMode(13, OUTPUT)** artinya kita menentukan pin digital 13 pada Arduino berfungsi sebagai output.

2) **digitalWrite(pin, VAL)**

Fungsi ini digunakan untuk menuliskan nilai secara digital pada suatu pin. Nilai VAL dapat berupa HIGH (ON) atau LOW (OFF) dan nilai pin adalah nomor pin pada Arduino/ESP yang akan diset. Contoh: **digitalWrite(13, HIGH)** artinya pin digital 13 diset pada kondisi menyala.

3) **digitalRead(pin)**

Fungsi ini digunakan untuk membaca nilai input/masukan yang diberikan ke Arduino/ESP. Nilai yang terbaca oleh Arduino melalui **digitalRead()** bergantung pada voltase pada pin yang diatur. Kebergantungan pada nilai voltase ini disebut Logic Level. Batasan nilai yang mencukupi untuk mencapai HIGH adalah di antara 5-3 volt, sedangkan batasan nilai yang mencapai nilai LOW adalah di antara 0 1,5 volt. Contoh: **digitalRead(13)** artinya Arduino akan membaca input yang diberikan melalui pin 13, hasilnya HIGH atau LOW.

4) **analogWrite(pin, VAL)**

Fungsi **analogWrite()** adalah fungsi yang digunakan untuk menuliskan suatu nilai berupa angka pada sebuah komponen, misalnya LED. Pengguna dapat mengatur seberapa terang cahaya dari lampu LED saat menyala, tergantung pada nilai yang dituliskan. Fungsi ini akan berguna ketika kita mulai bermain dengan sensor, di mana nilai yang terbaca seringkali berupa analog (memiliki banyak nilai, misal 0-1023), bukan digital (hanya memiliki 2 nilai, 0 (LOW) dan 1(HIGH)). Contoh: **analogWrite(13, 1023)** artinya Arduino akan memberikan instruksi nilai maksimal pada pin 13, sehingga komponen yang terkoneksi di pin 13 akan menyala maksimal (1023).

5) **analogRead(pin)**

Fungsi ini mirip dengan fungsi **digitalRead()**, yaitu membaca nilai masukan pada suatu pin. Bedanya adalah fungsi **analogRead()** akan menghasilkan nilai dari 0 hingga 1023, yang merepresentasikan voltase 0 v hingga 5 v. Contoh: **analogRead(0)** artinya Arduino akan membaca nilai input dari pin 0.

6) **delay(time)**

Fungsi ini digunakan untuk memberikan jeda antar fungsi. Nilai time adalah waktu lamanya jeda dalam satuan ms (milisekon), di mana 1 detik setara dengan 1.000 milisekon.

7) **Serial.begin(baudrate)**

Pengguna dapat melakukan komunikasi serial antara Microcontroller dengan PC, dengan menggunakan Serial Monitor yang disediakan pada Arduino IDE. Pada Serial Monitor, kita bisa melihat data yang dikirim dari Microcontroller ke PC. Selain itu, kita juga bisa mengirim data ke Arduino dengan cara mengetikkannya pada textbox di bagian atas Arduino IDE. Untuk memakai serial, yang pertama harus dilakukan adalah melakukan inisiasi dengan menggunakan fungsi **Serial.begin(baudrate)**. Variabel baudrate merupakan rasio modulasi, dan harus dicocokkan dengan baudrate hardware yang akan dikomunikasikan. Contoh: **Serial.begin(9600)** artinya komunikasi akan berjalan pada rasio modulasi/baudrate 9600.

8) **Serial.available()**

Fungsi ini digunakan untuk mengetes apakah ada input data dari hardware yang disambungkan ke serial port, misalnya dari PC. Fungsi ini akan menghasilkan 1 apabila ada masukan, dan 0 apabila tidak ada masukan.

9) **Serial.read()**

Fungsi ini berfungsi untuk membaca karakter pada serial port. Karakter yang dibaca akan disimpan dalam bentuk ASCII (misalnya karakter '0' memiliki representasi ASCII yaitu 48).

10) **Serial.print()** dan **Serial.println()**

Fungsi ini digunakan untuk menuliskan suatu kalimat ke Serial Monitor, tetapi tidak mengirimkan data apapun, alias hanya digunakan untuk memberikan teks visual pada pengguna. **Serial.print("text")** digunakan untuk menulis "text", sedangkan **Serial.println("text")** dipakai untuk menuliskan kata "text" dan diakhiri dengan enter (kalimat selanjutnya ada di baris berikutnya).

11) Serial.write(VAL)

Untuk mengirimkan data dari Microcontroller ke PC, kita bisa menggunakan fungsi **Serial.write(VAL)**. Nilai VAL adalah data yang ingin dikirimkan dari arduino ke PC, dengan ukuran 1 byte.

2. Variabel dan Tipe Data

Variabel adalah tempat untuk menyimpan sebuah nilai, dan tipe data adalah jenis nilai yang akan disimpan di dalam sebuah variabel. Untuk menulis sebuah variabel ada beberapa aturan dalam penulisannya, antara lain :

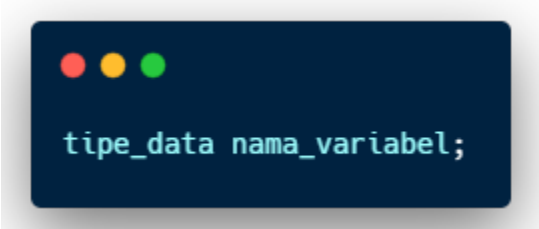
- Nama variabel tidak boleh didahului simbol dan angka.
- Nama variabel tidak boleh menggunakan kata kunci yang ada pada bahasa pemrograman seperti **if**, **int**, **float**, dll.
- Nama variabel bersifat *case sensitive*, artinya besar kecil huruf dibedakan, misalnya variabel **sensor** dengan **Sensor** adalah dua variabel yang berbeda.
- Jika nama variabel terdiri dari dua suku kata bisa menggunakan *underscore* atau digabungkan menjadi satu kata, misalnya **sensor_garis** atau **sensorgaris**.
- Saat pembuatan variabel diharuskan menambahkan tanda titik koma (;) di akhir baris.

Setiap Variabel juga memiliki jenis tipe datanya masing-masing, berikut adalah beberapa jenis tipe data yang sering digunakan.

Jenis Tipe Data	Bentuknya	Contoh
Integer (int)	Berupa kumpulan angka bilangan bulat positif atau negatif.	-1, 2, 0, 4, 10

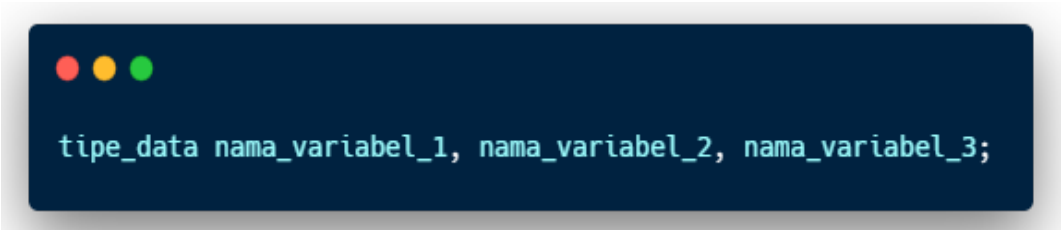
Float	Berupa kumpulan angka bilangan desimal positif atau negatif	(0.5), (12.6)						
Character (char)	Tipe data yang berupa sebuah karakter baik itu sebuah huruf, simbol ataupun spasi kosong	'A', '#', '-'						
Boolean (bool)	Tipe data yang hanya memiliki dua nilai yaitu benar dan salah, dalam bilangan biner direpresentasikan dengan nilai 1 dan 0	1 (true) 0 (false) Contoh : X = 1 Y = 4 A = X < Y Nilai X lebih kecil dari y maka A bernilai True						
String	Tipe data berupa teks yang bisa menyertakan angka, simbol atau karakter lain selama itu masih di dalam tanda kutip	"Hello World" "Robotic School 2022" "admin@gmail.com"						
Array	Tipe data yang berupa sejumlah elemen dari urutan tertentu. Dan tiap total elemen array mewakili panjang jenis data tersebut. Dan index array selalu dimulai dari 0	<table border="1"> <tr> <td>Semarang</td><td>Jakarta</td><td>Bandung</td></tr> <tr> <td>0</td><td>1</td><td>2</td></tr> </table> Data diatas merupakan array dari data nama kota yang memiliki panjang array 3.	Semarang	Jakarta	Bandung	0	1	2
Semarang	Jakarta	Bandung						
0	1	2						

Untuk membuat sebuah variabel dalam program harus dilakukan pendeklarasian. Format pendeklarasian sebuah variabel adalah sebagai berikut :



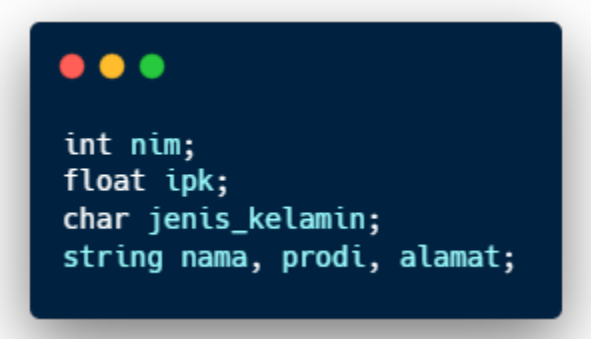
```
tipe_data nama_variabel;
```

Variabel-variabel yang memiliki tipe data yang sama dapat dideklarasikan dalam satu baris yang sama atau deklarasi tunggal sebagai sebuah daftar yang dipisahkan dengan tanda koma, contoh penulisannya adalah sebagai berikut :



```
tipe_data nama_variabel_1, nama_variabel_2, nama_variabel_3;
```

Contoh pendeklarasian variabel adalah sebagai berikut.

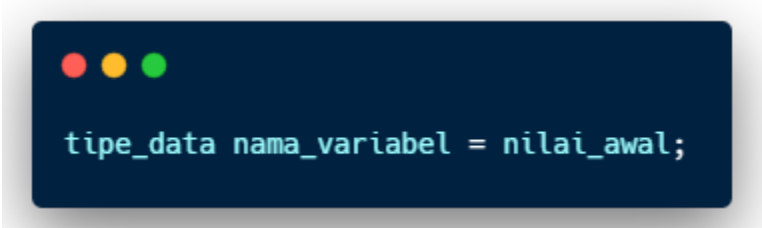


```
int nim;  
float ipk;  
char jenis_kelamin;  
string nama, prodi, alamat;
```

Pada contoh diatas pada baris pertama nama variabelnya adalah **nim** dengan jenis tipe data **integer**, kemudian pada baris kedua ada variabel **ipk** dengan tipe data **float**, dan pada baris ketiga karena nama variabel terdiri dari dua suku kata, maka menggunakan *underscore* untuk memisahkannya sehingga nama variabelnya menjadi **jenis_kelamin** dan pada baris terakhir karena beberapa variabel memiliki tipe data yang sama, maka bisa dituliskan dalam satu baris yang sama dipisahkan dengan tanda

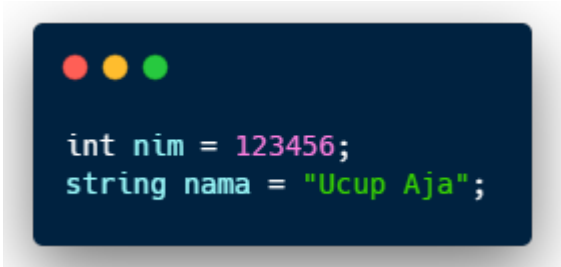
koma (,) . Contoh diatas adalah pendeklarasian variabel. Lalu bagaimana memberikan sebuah nilai ke dalam sebuah variabel?.

Untuk memberikan/menyimpan nilai awal ke dalam variabel disebut inisialisasi. Pengisian nilai dilakukan dengan menambahkan operator sama dengan (=). Format penulisannya adalah sebagai berikut :



```
tipe_data nama_variabel = nilai_awal;
```

Contoh inisialisasi variabel :



```
int nim = 123456;  
string nama = "Ucup Aja";
```

Contoh diatas merupakan inisialisasi terhadap variabel **nim** dengan nilai awal 123456 dan variabel **nama** dengan nilai awal "Ucup Aja".

3. Operator

Dalam bahasa pemrograman ada yang dinamakan operator, operator berguna untuk membentuk hasil nilai yang diinginkan. Sederhananya operator adalah simbol yang digunakan dalam penulisan bahasa pemrograman untuk menentukan sesuatu.

Ada beberapa jenis operator, antara lain :

- Operator Aritmatika

Operator	Jenis Operasi	Keterangan
+	Penjumlahan	Digunakan dalam penjumlahan
-	Pengurangan	Digunakan dalam pengurangan
*	Perkalian	Digunakan dalam perkalian
/	Pembagian	Digunakan dalam pembagian
%	Sisa Bagi	Untuk menentukan sisa dari hasil bagi Contohnya $a = 7\%2$ Operasi diatas hasilnya adalah 1, karena 7 dibagi dengan 2 dan menghasilkan sisa 1 dan nilai 1 inilah yang disebut sisa hasil bagi

- Operator Relasional

Operator	Jenis Operasi	Contoh
>	Lebih besar	$4 > 2$
<	Lebih kecil	$1 < 3$
>=	Leih besar atau sama dengan	$3 >= 3$
<=	Lebih kecil atau sama dengan	$2 <= 2$
==	Sama dengan	$5 == 5$
!=	Tidak sama dengan	$4 != 2$

- **Operator Bitwise**

Operator	Jenis Operasi	Contoh
&	AND	1 & 0 = 0
 	OR	1 0 = 1
^	XOR	1 ^ 1 = 0
~	NOT	~1 = 0

- **Operator Logika**

Operator	Jenis Operasi	Contoh
&&	AND	1 && 1 = 1
 	OR	1 0 = 1
!	NOT	!0 = 1

4. Percabangan

Percabangan adalah perintah yang memungkinkan suatu statement dieksekusi jika suatu kondisi terpenuhi atau tidak terpenuhi. Jika kondisinya terpenuhi maka perintah/statement akan dijalankan. Namun jika kondisinya tidak terpenuhi maka perintah/statement lain yang akan dijalankan.

Percabangan biasanya digunakan untuk menentukan langkah kerja instruksi menggunakan operator kondisional yang akan menghasilkan nilai boolean yaitu benar atau salah. Ada beberapa jenis percabangan yang ada di pemrograman yaitu.

- **if... / if... else... / if... else if... else...**

Struktur (**if...**) dibentuk untuk menyeleksi suatu kondisi dan statement tunggal sedangkan (**if... else...**) untuk menyeleksi satu kondisi dengan statement lebih dari satu dan (**if... else if... else...**) untuk menyeleksi beberapa kondisi dengan statement lebih dari satu. Kondisi digunakan untuk menentukan

pengambilan keputusan sedangkan statement berupa pernyataan tunggal atau majemuk yang akan dijalankan ketika suatu kondisi terpenuhi.

- Pada percabangan **if**, bila kondisi terpenuhi maka statement yang ada di dalam blok if akan diproses atau dijalankan. Bentuk struktur dari percabangan if adalah sebagai berikut :

A code editor window with a dark blue background and three colored window control buttons (red, yellow, green) at the top left. It contains the following code:


```
if(kondisi1){  
    Statement_1;  
}
```

- Pada percabangan **if else**, jika kondisi bernilai benar maka akan menjalankan statement pertama, dan jika kondisi salah maka akan menjalankan statement yang kedua. Bentuk struktur dari percabangan if else adalah sebagai berikut :

A code editor window with a dark blue background and three colored window control buttons (red, yellow, green) at the top left. It contains the following code:

```
if(kondisi1){  
    Statement_1;  
}  
else{  
    Statement_2;  
}
```

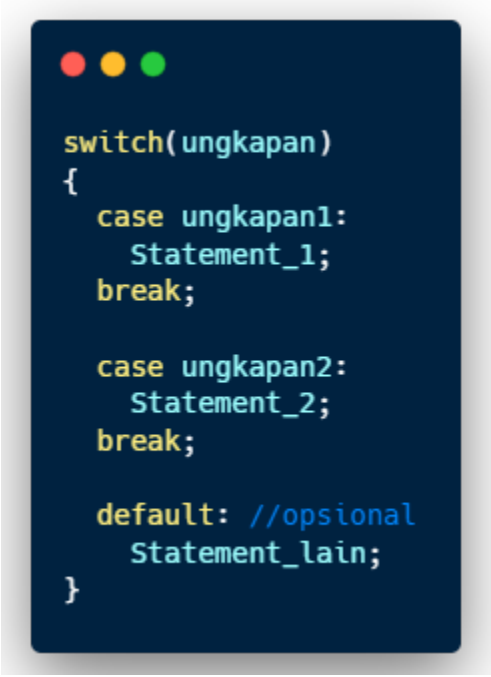
- Dan pada percabangan **if.. else if... else...**, bila kondisi bernilai benar maka akan menjalankan statement yang ada di blok **if** kondisi pertama, namun jika kondisi pertama salah maka akan melakukan pengecekan pada kondisi kedua. Jika kondisi kedua benar maka akan menjalankan statement pada blok **else if** kondisi kedua, namun jika salah maka akan menjalankan statement di blok **else**. Bentuk struktur dari percabangan **if... else if... else...** adalah sebagai berikut :



```
if(kondisi1){  
    Statement_1;  
}  
else if(kondisi2){  
    Statement_2;  
}  
else{  
    Statement_3;  
}
```

- **switch case**

Percabangan switch case digunakan untuk menjalankan salah satu statement dari beberapa kemungkinan pernyataan, berdasarkan nilai dari sebuah nilai ungkapan dan nilai penyeleksi. Bentuk struktur dari blok percabangan **switch case** adalah sebagai berikut :

A code editor window with a dark blue background and three colored window control buttons (red, yellow, green) at the top left. It contains a C-style switch case code snippet with syntax highlighting: 'switch' and 'break' are in yellow, 'case' and 'default' are in light blue, and the variable 'ungkapan' and its cases are in white. The code is as follows:

```
switch(ungkapan)
{
    case ungkapan1:
        Statement_1;
        break;

    case ungkapan2:
        Statement_2;
        break;


    default: //opsional
        Statement_lain;
}
```

5. Perulangan/Looping

Perulangan atau looping adalah bentuk program untuk menuliskan/menjalankan instruksi yang sama berulang-ulang dan akan berhenti sampai kondisi tertentu. Perulangan ada beberapa jenisnya, yaitu :

- FOR

Perulangan FOR dilakukan dengan cara mengulangi instruksi berdasarkan ekspresi yang diberikan. Pengulangan FOR digunakan ketika ingin melakukan perulangan yang batas pengulangannya sudah diketahui. Bentuk penulisan pada blok perulangan FOR adalah sebagai berikut :

A screenshot of a code editor with a dark blue background and three colored window control buttons (red, yellow, green) in the top left corner. The code is written in a light blue/cyan monospace font. It shows the syntax for a FOR loop:

```
for (<ekspresi1>;<ekspresi2>;<ekspresi3>)  
{  
    Statement;  
}
```

- ekspresi 1 : inisialisasi nilai awal perulangan
- ekspresi 2 : kondisi kapan perulangan berhenti
- ekspresi 3 : pengatur kenaikan nilai variabel loop

- DO.. WHILE

DO.. WHILE dilakukan dengan melakukan pengecekan kondisi diakhir setelah pernyataan/statement dijalankan. Sehingga dalam perulangan DO.. WHILE akan ada minimal satu kali proses pengulangan walau kondisi bernilai salah. Bentuk penulisan pada blok perulangan DO.. WHILE adalah sebagai berikut :



```
do{  
    Statement;  
}while(kondisi);
```

Dalam penulisan blok DO.. WHILE, kondisi akan ditulis setelah pernyataan karena di perulangan jenis ini pernyataan akan dijalankan terlebih dahulu sebelum kemudian dilakukan pengecekan kondisi benar atau salah, jika benar maka akan dilakukan perulangan/menjalankan pernyataan kembali dan akan berhenti sampai kondisi bernilai salah.

- WHILE

Perulangan WHILE akan melakukan perulangan jika kondisi terpenuhi atau bernilai benar dan akan berhenti ketika kondisi tidak terpenuhi atau bernilai salah. WHILE akan melakukan pengecekan kondisi terlebih dahulu sebelum menjalankan statement yang ada di dalam blok perulangan. Bentuk penulisan pada blok perulangan WHILE adalah sebagai berikut :

A dark-themed code editor window with three colored window control buttons (red, yellow, green) at the top left. The code inside is written in a light blue/cyan font and shows the syntax for a while loop:

```
while(kondisi){  
    Statement;  
}
```

B. ARDUINO IDE

Arduino IDE (Integrated Development Environment) merupakan software atau media untuk menuliskan program, mengcompile dan mengupload board microcontroller khususnya board arduino. Kode program yang digunakan di arduino disebut dengan istilah Arduino “Sketch” atau source code arduino dengan ekstensi file .ino.

1. Bagian-bagian Arduino IDE

a. Toolbar



- 1) **Verify**, Untuk memverifikasi terlebih dahulu sketch/program yang dibuat. Jika ada kesalahan pada sketch, akan muncul error.
- 2) **Upload**, Untuk mengupload sketch ke board Arduino. Jika kita tidak mengklik tombol verify, maka sketch akan di-compile, kemudian langsung di upload ke board.
- 3) **New Sketch**, Membuka window dan membuat sketch baru.
- 4) **Open Sketch** Membuka sketch yang sudah pernah dibuat dengan ekstensi file .ino
- 5) **Save Sketch**, Untuk menyimpan program.
- 6) **Serial Monitor** Membuka interface untuk komunikasi serial. Di serial monitor akan menunjukkan data yang dipertukarkan antara arduino dan komputer selama beroperasi, sehingga kamu bisa menggunakan serial monitor ini untuk menampilkan nilai hasil operasi atau pesan debugging. Selain melihat data, serial monitor juga bisa mengirimkan data ke Arduino dengan memasukkan data pada text box dan menekan tombol send untuk mengirimkan data. Hal penting yang harus perhatikan adalah menyamakan baudrate antara serial monitor dengan Arduino board. Untuk menggunakan serial monitor pada Arduino, di bagian fungsi void setup(), diawali dengan instruksi **Serial.begin** diikuti nilai baudrate.

b. Menu Bar

- **File**

- 1) **New**, Untuk membuat membuat sketch baru.
- 2) **Open**, Untuk membuka sketch yang pernah dibuat.
- 3) **Open Recent**, Untuk membuka file atau sketsa yang baru-baru ini sudah dibuat.
- 4) **Sketchbook**, Untuk menunjukan hirarki sketch yang dibuat termasuk struktur foldernya.
- 5) **Example**, berisi contoh-contoh pemrograman yang sudah disediakan oleh pengembang arduino yang bisa dipelajari.
- 6) **Close**, Untuk menutup jendela Arduino IDE dan menghentikan aplikasi.
- 7) **Save**, Untuk menyimpan sketch yang dibuat.
- 8) **Save as...**, Untuk menyimpan sketch yang sedang dikerjakan dengan nama yang berbeda.
- 9) **Page Setup**, Untuk mengatur tampilan page pada proses pencetakan.
- 10) **Print**, Untuk mengirimkan file sketch ke mesin cetak untuk dicetak.
- 11) **Preferences**, Untuk merubah tampilan interface IDE Arduino.
- 12) **Quit**, Untuk menutup semua jendela Arduino IDE.

- **Edit**

- 1) **Undo/Redo**, Untuk mengembalikan perubahan yang sudah dilakukan pada Sketch.
- 2) **Cut**, Untuk meremove teks yang dipilih dan menempatkan teks tersebut pada clipboard.
- 3) **Copy**, Untuk menduplikasi/mengcopy teks yang terpilih.
- 4) **Copy for Forum**, Untuk copy kode dan melakukan formating untuk ditampilkan dalam forum.
- 5) **Copy as HTML**, Untuk mengcopy teks yang terpilih dan meletakkannya di clipboard dalam bentuk atau format HTML.
- 6) **Paste**, Untuk menyalin data yang ada pada clipboard, kedalam editor.
- 7) **Select All**, Untuk melakukan pemilihan teks atau kode dalam halaman.

- 8) **Comment/Uncomment**, Untuk memberikan atau menghapus tanda // pada kode, yang menjadikan suatu baris kode sebagai komen dan tidak akan dijalankan ketika tahap kompilasi.
- 9) **Increase/Decrease Indent**, Untuk mengurangi dan menambahkan indentasi pada baris kode.
- 10) **Find**, Untuk memanggil jendela window find and replace, untuk menemukan kata dalam program atau menemukan dan menggantinya (replace) kata yang dipilih dengan kata lain.
- 11) **Find Next**, berfungsi menemukan kata selanjutnya dari kata pertama yang sudah ditemukan.
- 12) **Find Previous**, berfungsi menemukan kata sebelumnya dari kata pertama yang sudah ditemukan.

- **Sketch**

- 1) **Verify/Compile**, Untuk mengcompile program, jika terdapat kesalahan maka akan menampilkan pesan error atau kesalahan.
- 2) **Upload**, Untuk mengupload atau mengirimkan program ke Arduino Board.
- 3) **Upload Using Programmer**, Untuk menuliskan bootloader kedalam IC Mikrokontroler Arduino. Membutuhkan perangkat tambahan seperti USBAsp sebagai perantara penulisan program bootloader ke IC Mikrokontroler.
- 4) **Export Compiled Binary**, Untuk menyimpan file dengan ekstensi .hex, agar dapat disimpan sebagai arsip dan di upload ke board lain dengan tools yang berbeda.
- 5) **Show Sketch Folder**, Untuk membuka folder sketch yang saat ini dikerjakan.
- 6) **Include Library**, Untuk menambahkan library/pustaka kedalam sketch dengan menyertakan sintaks #include di awal program. Dan juga bisa menambahkan library eksternal dari file .zip ke Arduino IDE.

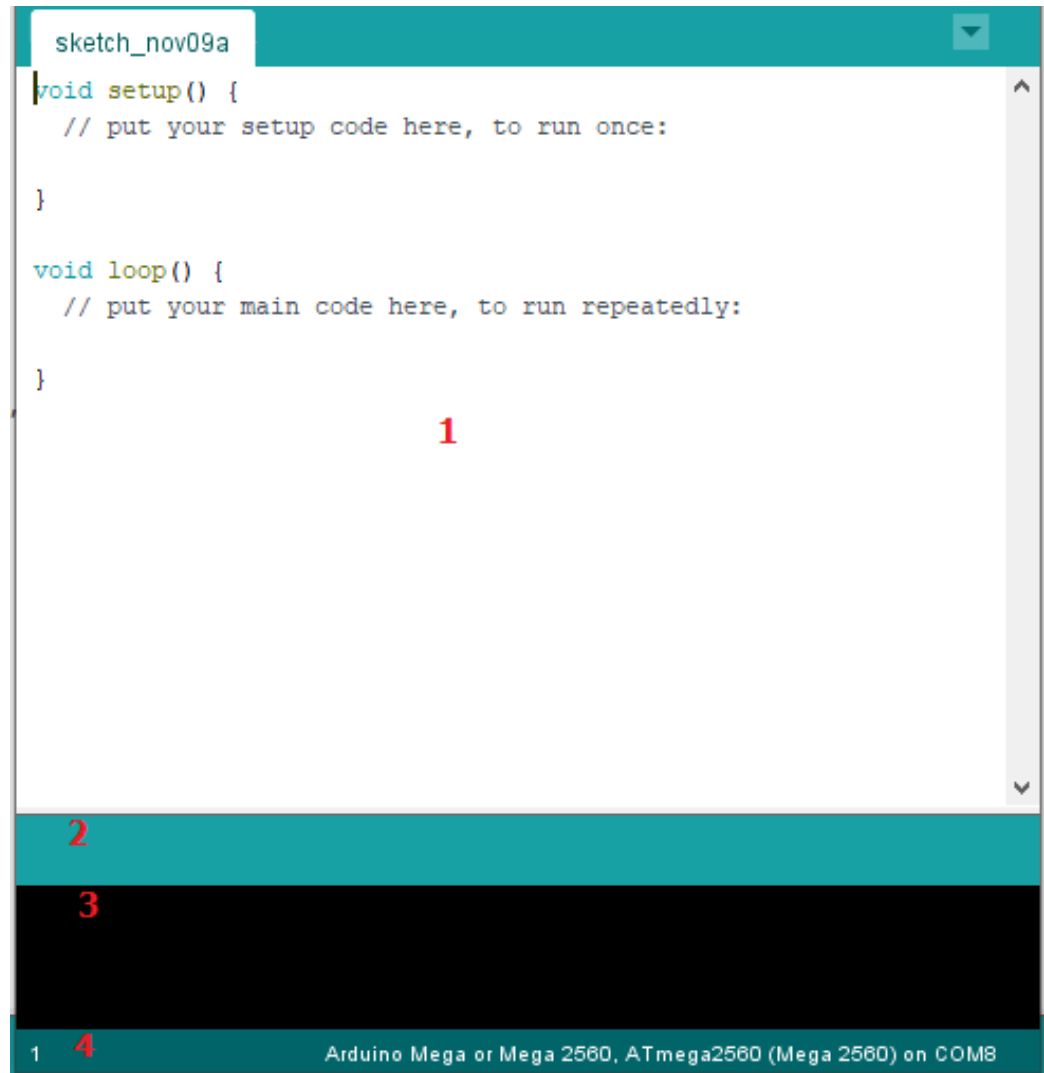
- 7) **Add File...**, Untuk menambahkan file kedalam sketch arduino, file yang ditambahkan akan muncul sebagai tab baru di jendela sketch.

- **Tools**

- 1) **Auto Format**, Untuk melakukan pengaturan format kode pada jendela editor.
- 2) **Archive Sketch**, Untuk menyimpan sketch kedalam file .zip
- 3) **Fix Encoding & Reload**, Untuk memperbaiki kemungkinan perbedaan antara pengkodean peta karakter editor dan peta karakter sistem operasi yang lain.
- 4) **Serial Monitor**, Untuk membuka jendela serial monitor untuk melihat pertukaran data.
- 5) **Board**, Untuk memilih dan konfigurasi board yang digunakan.
- 6) **Port**, Untuk memilih port sebagai media komunikasi antara software dengan hardware.
- 7) **Programmer**, menu ini digunakan saat melakukan pemrograman chip mikrokontroler tanpa menggunakan koneksi Onboard USB-Serial. Biasanya saat proses burning bootloader.
- 8) **Burn Bootloader**, Untuk mengkopikan program bootloader kedalam IC mikrokontroler.

- **Help**

- 1) Berisikan file-file yang berhubungan dengan masalah yang sering muncul, serta solusinya. Pada menu help juga diberikan link untuk menuju Arduino Forum untuk menanyakan dan berdiskusi tentang masalah yang ditemukan.



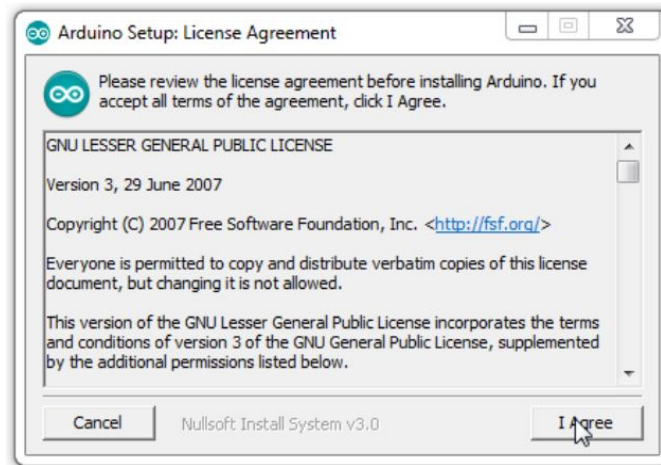
- 1) Workspace, merupakan tempat dimana kode program ditulis, terdapat dua fungsi yaitu fungsi *Void setup()* adalah fungsi yang menjalankan program hanya satu kali dan *Void loop()* adalah fungsi yang akan menjalankan program berulang-ulang atau terus menerus sampai power dinonaktifkan.
- 2) **Keterangan Aplikasi**, akan memunculkan pesan-pesan yang hasil operasi, misal Compiling dan Done Uploading ketika kita sedang compile dan mengupload sketch ke board Arduino
- 3) **Console log**, sama dengan Keterangan Aplikasi hanya saja pesan yang ditampilkan lebih detail. Misal, ketika aplikasi meng compile atau ketika

ada kesalahan pada sketch yang kita buat, maka informasi error dan baris akan diinformasikan di bagian ini.

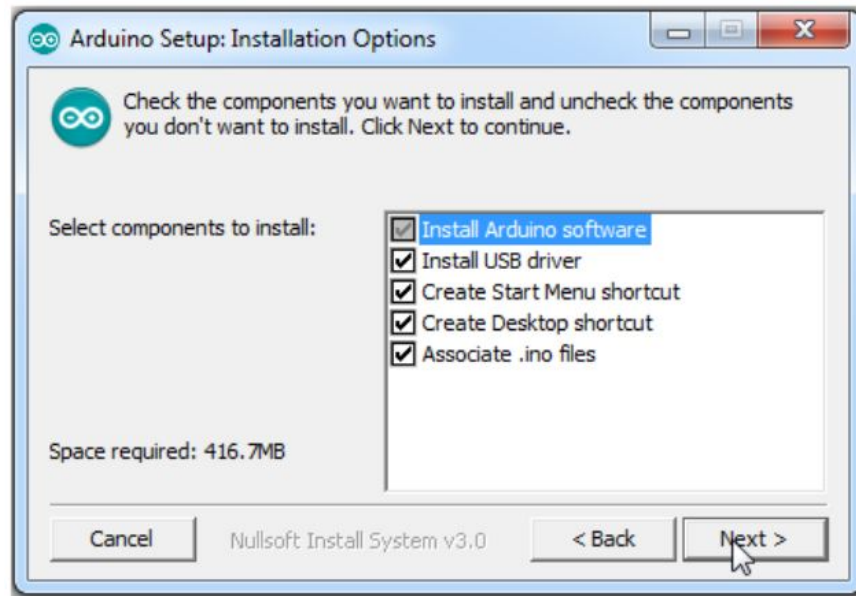
- 4) **Baris Sketch**, pada bagian kiri akan menunjukkan posisi baris kursor yang sedang aktif dan pada sketch bagian kanan adalah Informasi Board dan Port Bagian ini menginformasikan port yang dipakai oleh board Arduino.

2. LANGKAH LANGKAH INSTAL ARDUINO IDE

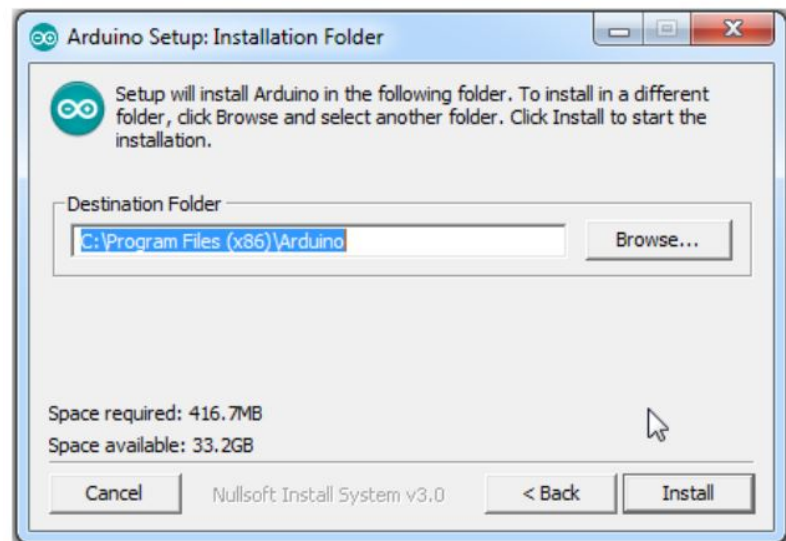
- a. Buka software IDE Arduino yang sudah di download dengan klik kiri dua kali atau klik kanan open. Akan muncul **License Agreement** atau **Persetujuan Instalasi**, klik **I Agree** untuk memulai install software program IDE Arduino.



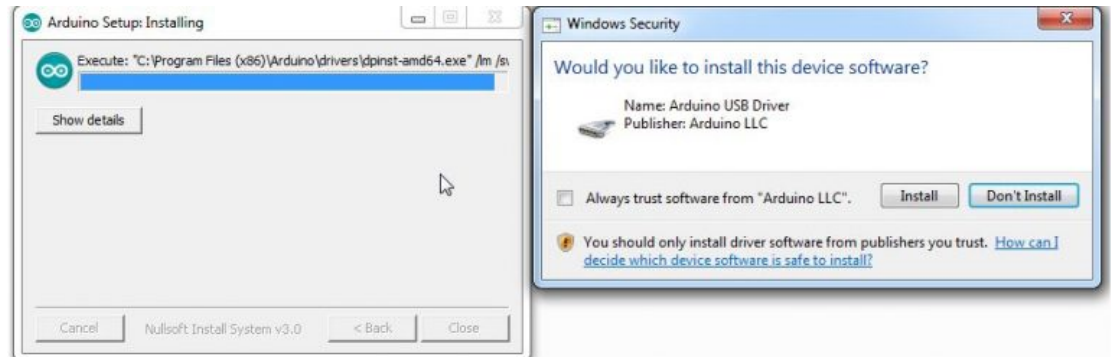
- b. Kemudian pada pilihan opsi instalasi / **Installation Options** pilih semua option dan klik tombol **Next**.



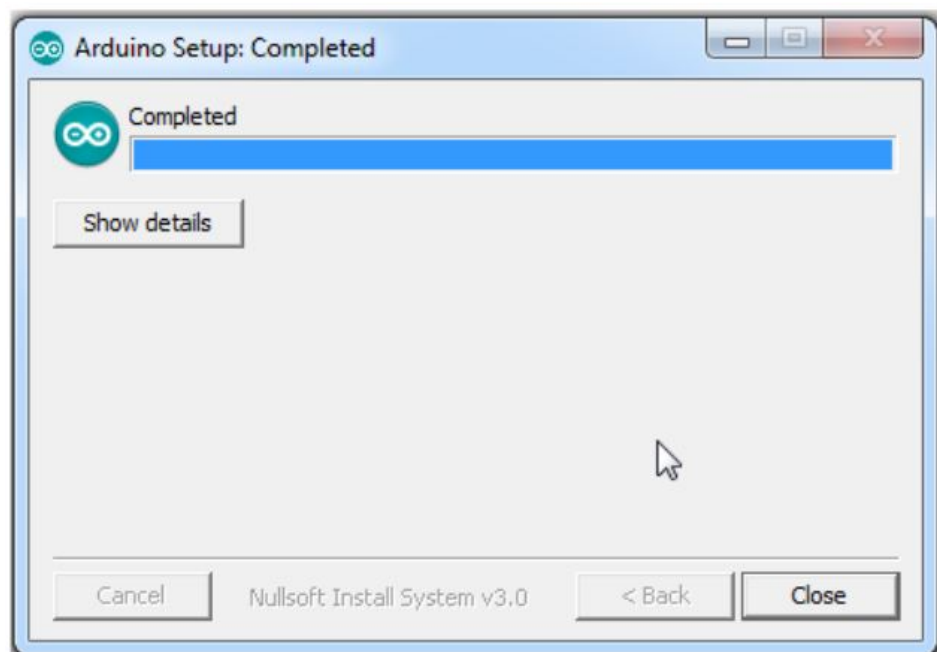
- c. Selanjutnya pada **Installation Folder** atau **Pilihan Folder**, pilih folder yang akan dijadikan tempat menyimpan file arduino. Kemudian klik tombol **Install** untuk memulai proses instalasi.



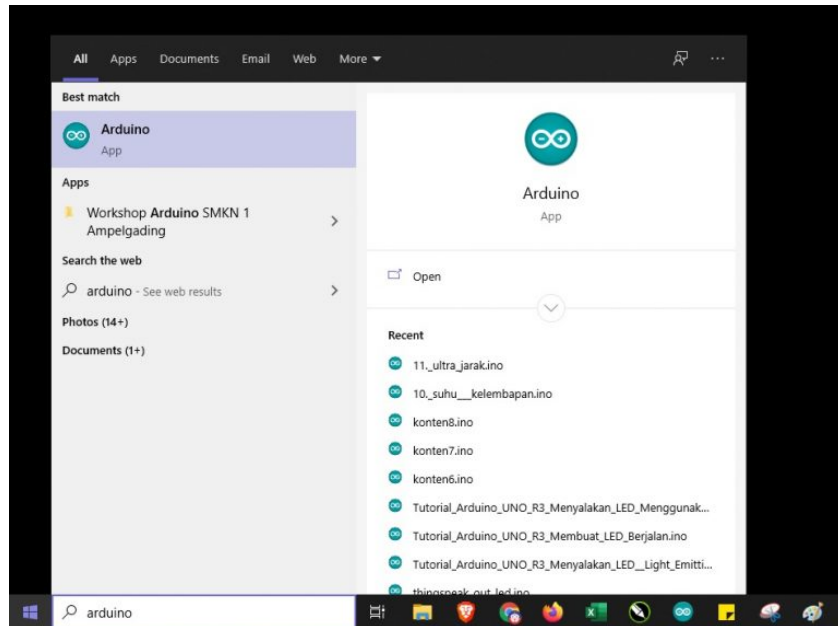
- d. Selanjutnya ketika proses instalasi sedang berjalan akan muncul pilihan untuk install driver, pilih tombol **Install**. Proses ini berfungsi untuk mengenali dan melakukan komunikasi dengan board microcontroler yang disambungkan.



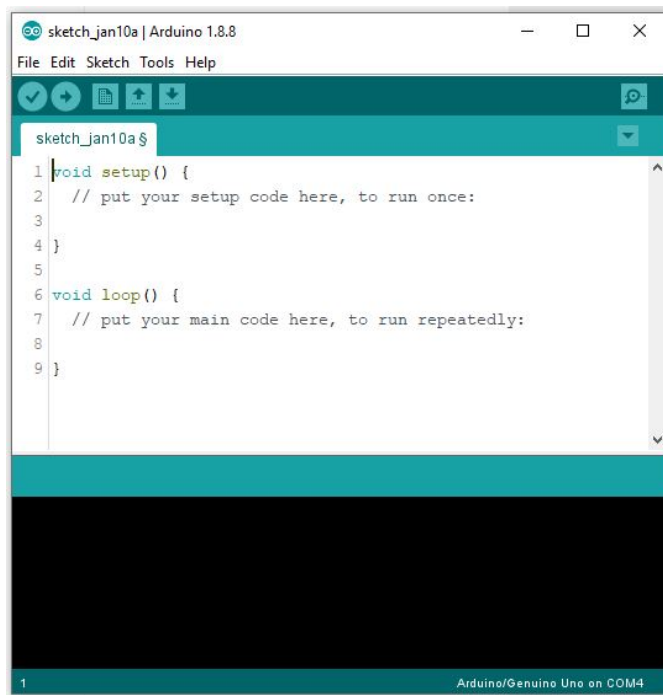
e. Setelah proses instalasi selesai klik close.



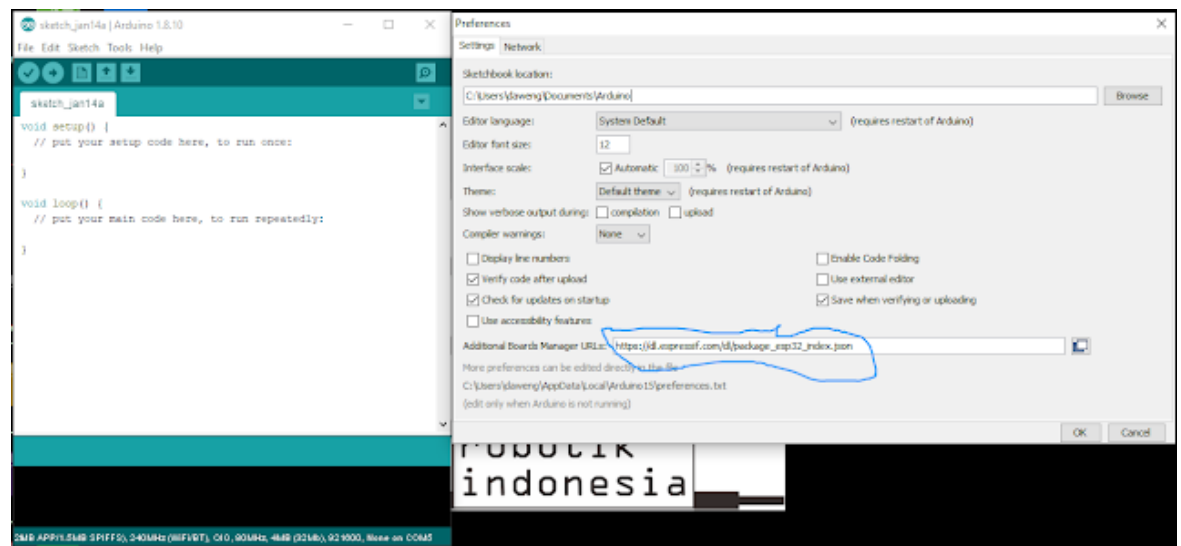
- f. Untuk menjalankan software arduino IDE, cek pada Start Menu pada bagian bawah kiri desktop windows, kemudian ketik arduino maka akan muncul software Arduino IDE. double klik icon Arduino untuk membukanya.



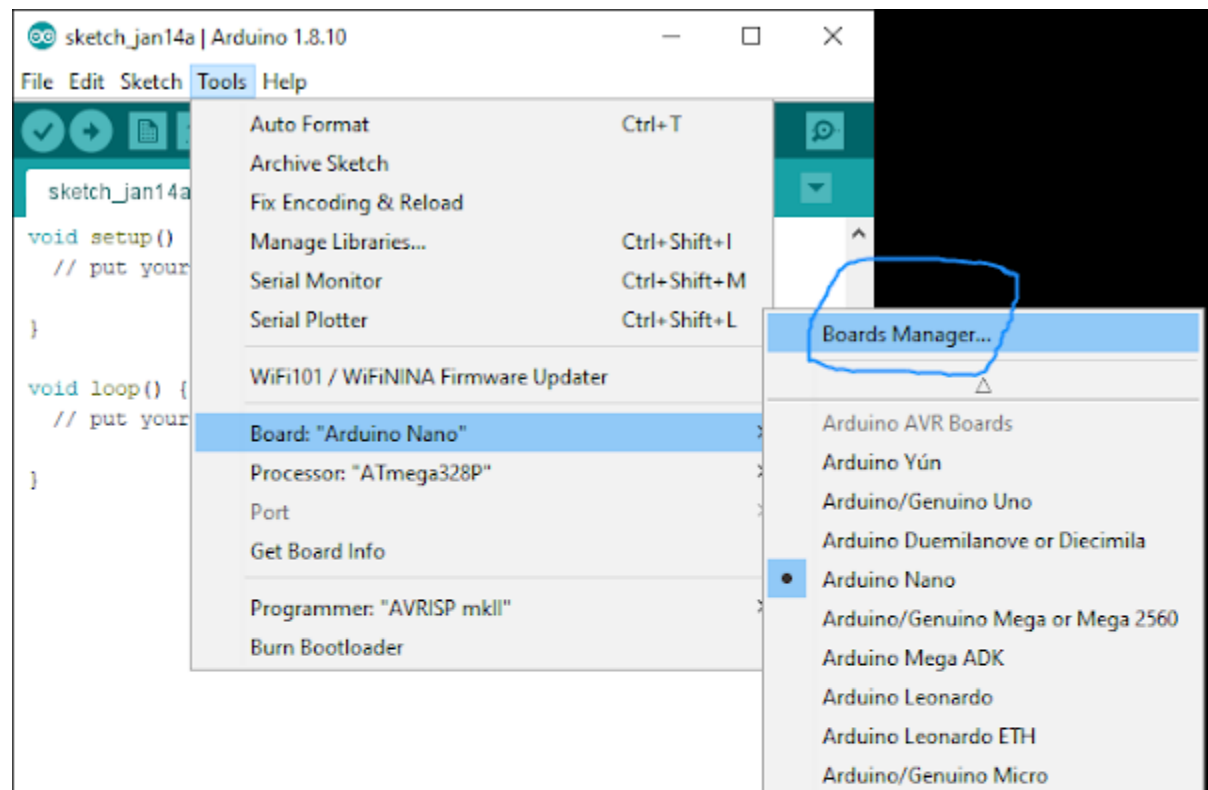
- g. Maka akan muncul jendela awal software Arduino IDE.



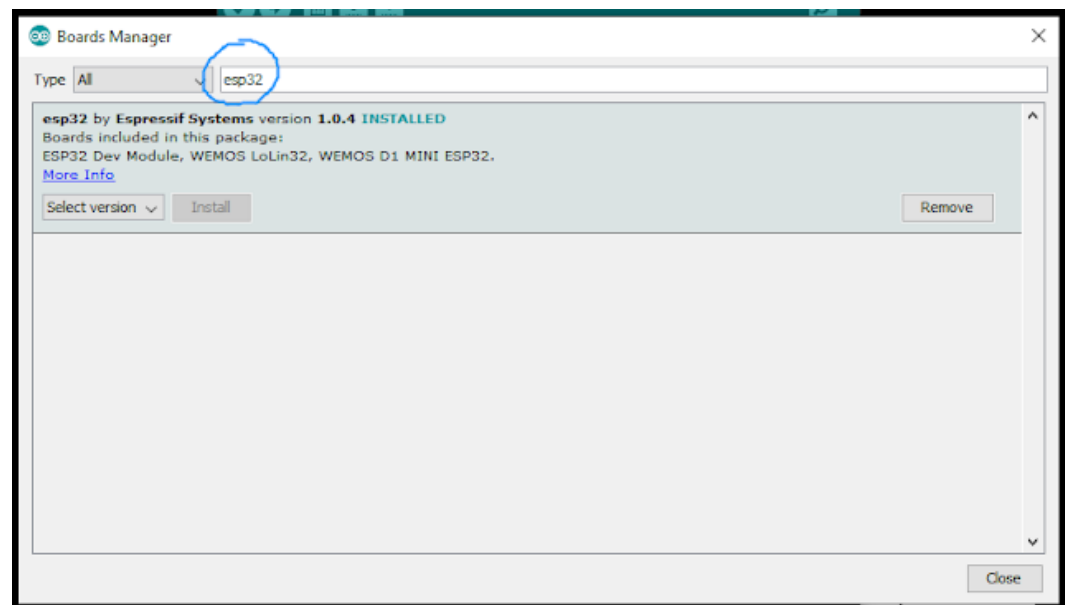
- h. Kemudian sebelum memulai memprogram microcontroller di Arduino IDE, pilih menu Tools lalu pilih Board dan pilih jenis microcontroller yang akan diprogram. Jika belum ada jenis board-nya pada daftar, bisa menginstalnya terlebih dahulu pada bagian Boards Manager, pada menu Tools > Board > Boards Manager.
- i. Untuk menambahkan board ESP, klik Menu File > Preferences. Pada kolom Additional ... yang ada dibawah, tambahkan link berikut
https://dl.espressif.com/dl/package_esp32_index.json
Kemudian OK



- j. Klik menu Tools > Board: > Pilih Boards Manager



- k. Pada kolom pencarian tulis ESP32 kemudian install dan tunggu sampai selesai.



LATIHAN PROGRAM ARDUINO DAN ESP32

LATIHAN 1

LED ON

- l. Hubungkan Arduino/ESP ke komputer/laptop dengan kabel USB
- m. Hubungkan 1 lampu LED pada Breadboard dengan menggunakan kabel jumper dan sambungkan pada pin 13.
- n. Buka software Arduino IDE di komputer/laptop
- o. Tulis kode berikut pada Workspace Arduino IDE :



```
int pinLed = 13;
void setup() {
    pinMode(pinLed,OUTPUT);
}
void loop() {
    digitalWrite(pinLed,HIGH);
}
```

- p. Pilih menu Tools dan pilih Port lalu klik COM5
- q. Klik Verify setelah tidak ada error klik Upload.


TUGAS

1. Buat rangkaian dengan 3 lampu LED yang menyala Bersamaan

LATIHAN II

FLIP FLOP

2. Hubungkan Arduino/ESP ke komputer/laptop dengan kabel USB
3. Hubungkan 2 lampu LED di Breadboard menggunakan kabel dan sambungkan pada pin 13 dan 12.
4. Buka software Arduino IDE
5. Tulis kode berikut pada Workspace Arduino IDE :



```
int pinLed1 = 13;
int pinLed2 = 12;
void setup() {

    pinMode(pinLed1,OUTPUT);
    pinMode(pinLed2,OUTPUT);

}
void loop() {

    digitalWrite(pinLed1,HIGH);
    delay(2000);
    digitalWrite(pinLed2, HIGH);

}
```

6. Pilih menu Tools dan pilih Port lalu klik COM5
7. Klik Verify setelah tidak ada error klik Upload.

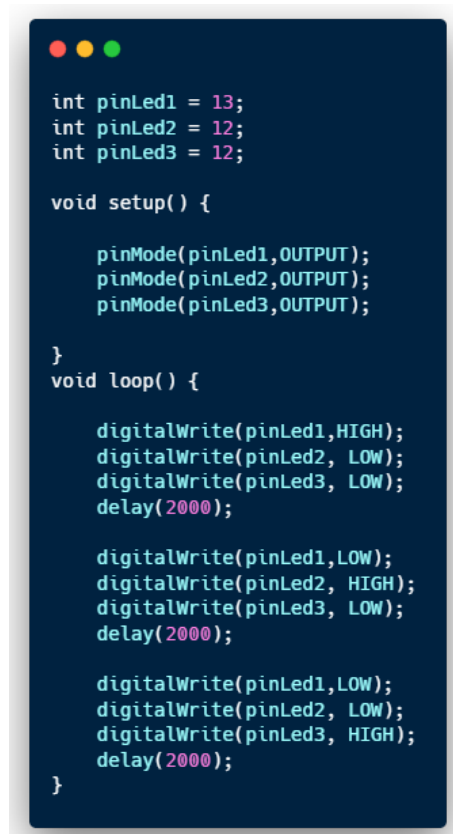
TUGAS

Buat rangkaian dengan 6 Lampu LED yang dapat menyala 2 lampu secara bergantian.

LATIHAN III

TRAFFIC LIGHT

1. Hubungkan Arduino/ESP ke komputer/laptop dengan kabel USB
2. Hubungkan 3 lampu LED dengan warna yang berbeda pada Breadboard menggunakan kabel jumper dan sambungkan pada pin 13, pin 12 dan pin 11.
3. Buka software Arduino IDE
4. Tulis kode berikut pada Workspace Arduino IDE :

A screenshot of the Arduino IDE workspace showing a C++ program for a traffic light simulation. The code is written in a dark-themed editor with syntax highlighting. It defines three LED pins (13, 12, 12) and configures them as outputs in the setup function. The loop function alternates the states of the LEDs in three steps, each with a 2000ms delay.

```
int pinLed1 = 13;
int pinLed2 = 12;
int pinLed3 = 12;

void setup() {

  pinMode(pinLed1,OUTPUT);
  pinMode(pinLed2,OUTPUT);
  pinMode(pinLed3,OUTPUT);

}
void loop() {

  digitalWrite(pinLed1,HIGH);
  digitalWrite(pinLed2, LOW);
  digitalWrite(pinLed3, LOW);
  delay(2000);

  digitalWrite(pinLed1,LOW);
  digitalWrite(pinLed2, HIGH);
  digitalWrite(pinLed3, LOW);
  delay(2000);

  digitalWrite(pinLed1,LOW);
  digitalWrite(pinLed2, LOW);
  digitalWrite(pinLed3, HIGH);
  delay(2000);

}
```

5. Pilih menu Tools dan pilih Port lalu klik COM5
6. Klik Verify setelah tidak ada error klik Upload.

TUGAS

Buatlah rangkaian dan program traffic light 2 jalur.