Lunguaggio

Arduino – Linguaggio

Il linguaggio di programmazione “C++” è un linguaggio ad alto livello, strutturato per poter sfruttare pienamente i microprocessori e i sistemi operativi attuali.

Arduino ne una un dialetto derivato da wired

Questo linguaggio, oggi, è basato sulla programmazione ad **oggetti** OOP (“C++”) ma è possibile impiegare la metodologia classica **procedurale** (con procedure e funzioni).

ARDUINO

Aldo Pizzeghella – C.S.E.

Arduino – Linguaggio

Un programma scritto in questo linguaggio, per essere eseguito, deve essere tradotto nel linguaggio macchina del processore che si impiega, questo passaggio si chiama **Compilazione**.

ARDUINO

Aldo Pizzeghella – C.S.E.

Arduino – Compilatore

Un **compilatore** è programma vero e proprio che ha il compito di tradurre ciò che scriviamo in codice comprensibile alla C.P.U.

Questo, durante la traduzione, può restituirci dei messaggi di errore, di avviso, di servizio con i quali siamo in grado di capire l'esito della traduzione.

All'interno del nostro programma è possibile inserire dei comandi che modificheranno il comportamento del compilatore (**direttive**) e altri comandi che verranno elaborati prima che il esso esegua il suo compito (**preprocessore**) questi ci aiutano semplificare la scrittura di programmi complessi.

ARDUINO

Aldo Pizzeghella – C.S.E.

Arduino – Linguaggio

Come tutti i linguaggi vi sono delle parole riservate alle quali è associato un specifico significato, per il resto tutto quello che si impiega dovrà essere dichiarato esplicitamente prima dell'uso.

Le istruzioni sono composte da un nome ed alcuni parametri (facoltativi) all'interno delle parentesi tonde:

```
istruzione(argomento, argomento, … argomento);
```

ARDUINO

Aldo Pizzeghella – C.S.E.

Arduino – Punteggiatura

Fine istruzione, ogni istruzione termina con questo carattere:

```
;
```

Commento singola riga tutto quello che segue questo simbolo verrà ignorato:

```
// bla bla bla
```

Commento su più righe:

```
/*  bla bla bla
bla bla bla
bla bla bla
*/
```

ARDUINO

Aldo Pizzeghella – C.S.E.

Arduino – Punteggiatura

Alcune istruzioni possono essere racchiuse tra parentesi graffe.
Il compilatore le considera un unico blocco, una unica istruzione:

```
{       istruzione();
        istruzione();
}
```

ARDUINO

Aldo Pizzeghella – C.S.E.

Arduino – Linguaggio: Procedure

Un blocco di istruzioni può avere un nome così sarà sufficiente richiamare l'intero blocco con il nome:

```
Nome_blocco() {
        istruzione(arg, arg, arg);
        istruzione(arg, arg, arg);
}
```

Questa si chiama **procedura**, e può ricevere dei valori dall'esterno:

```
Nome_procedura(argA, argB, argC,….) {
        istruzione(arg, arg, arg);
        istruzione(arg, arg, arg);
}
```

ARDUINO

Aldo Pizzeghella – C.S.E.

Arduino - Linguaggio

```
(void) setup()  {
        istruzione();
        istruzione();
}
```

Procedura verrà eseguita all'accensione del dispositivo

```
(void) loop()  {
        istruzione();
        istruzione();
}
```

Questa sarà eseguita successivamente al setup().

La procedura verrà ripetuta all'infinito, fino a quando non verrà spento il dispositivo

ARDUINO

Aldo Pizzeghella – C.S.E.

Arduino – Operazioni sui PIN digitali

pinMode(numero_pin, modalità);

imposta il pin (0..13) specificato la modalità desiderata (setup)
modalità: **OUTPUT** - **INPUT** - **INPUT_PULLUP**

digitalWrite(numero_pin, status);

se il pin (0..13) è impostato in OUTPUT cambia il suo valore in uscita in:
status: **LOW** - **0**: 0 Volt - segnale basso
HIGH -**1**: 5 Volt - segnale alto

(status) digitalRead(numero_pin);

Funzione che restituisce il valore del pin (0..13) se impostato in INPUT
status: **LOW** - **0**: 0 Volt - segnale basso
HIGH - **1**: 5 Volt - segnale alto

ARDUINO

Aldo Pizzeghella – C.S.E.

Arduino - Linguaggio

delay(tempo);

Esegue una attesa di (tempo) in millisecondi prima di proseguire
tempo: **10000**

ARDUINO

Aldo Pizzeghella – C.S.E.

Arduino – Suono

tone(pin, frequenza, durata);
Genera in uscita sul **pin** specificato una **frequenza** (Hz – tono)
per la **durata** (mS)
La durata è facotativa se la si omette il tono è continuo.

noTone(pin);
Interrompe la frequenza che genera.

ARDUINO

Aldo Pizzeghella – C.S.E.

Arduino – Terminale: Porta seriale

Serial.begin(velocità);
imposta la velocità della porta seriale (bit per secondo)
velocità: 2400 – 4800 – **9600 - 115200**

Serial.print(dato da inviare);
Serial.println(dato da inviare);
Invia sulla porta seriale l'informazione

(value) Serial.read();
Legge e restituisce ciò che viene ricevuto dalla porta seriale.

ARDUINO

Aldo Pizzeghella – C.S.E.

Arduino – Linguaggio: Procedure

```
void accendiLED() {
   digitalWrite(13, HIGT);
}

void lampeggioLED(int pin) {
   digitaWrite(pin,HIGT);
   delay(2000);
   digitaWrite(pin, LOW);
}
```

```
void loop() {
...
…
   accendiLED();
   lampeggioLED(10);
…
...
   lampeggioLED(5);
}
```

ARDUINO

Aldo Pizzeghella – C.S.E.

Arduino – Linguaggio: Funzioni

Alcune procedure possono restituire un valore come esito della loro attività, queste si chiamano **funzioni**:

```
(valore) nomeblocco(argA, argB, argC,….) {
        istruzione(arg, arg, arg);
        istruzione(arg, arg, arg);
        return(valore);
}
```

ARDUINO

Aldo Pizzeghella – C.S.E.