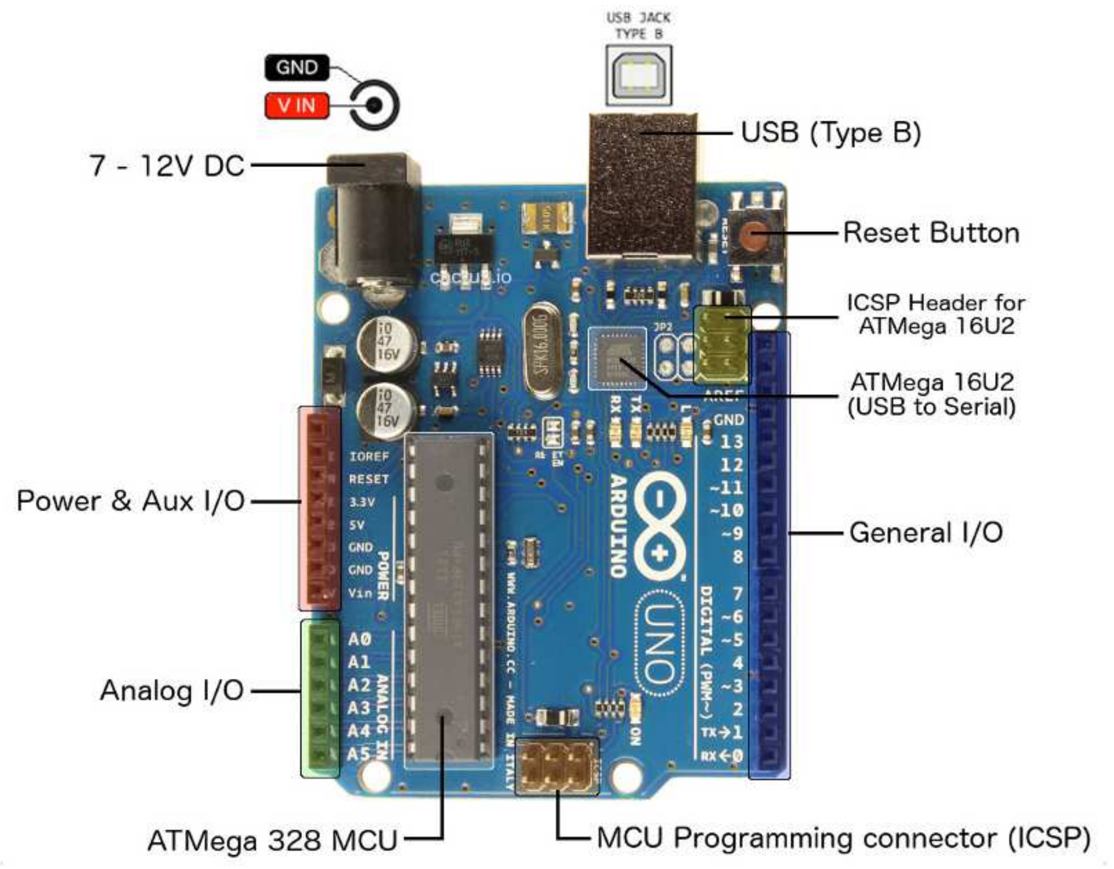
Arduino Uno pin diagram

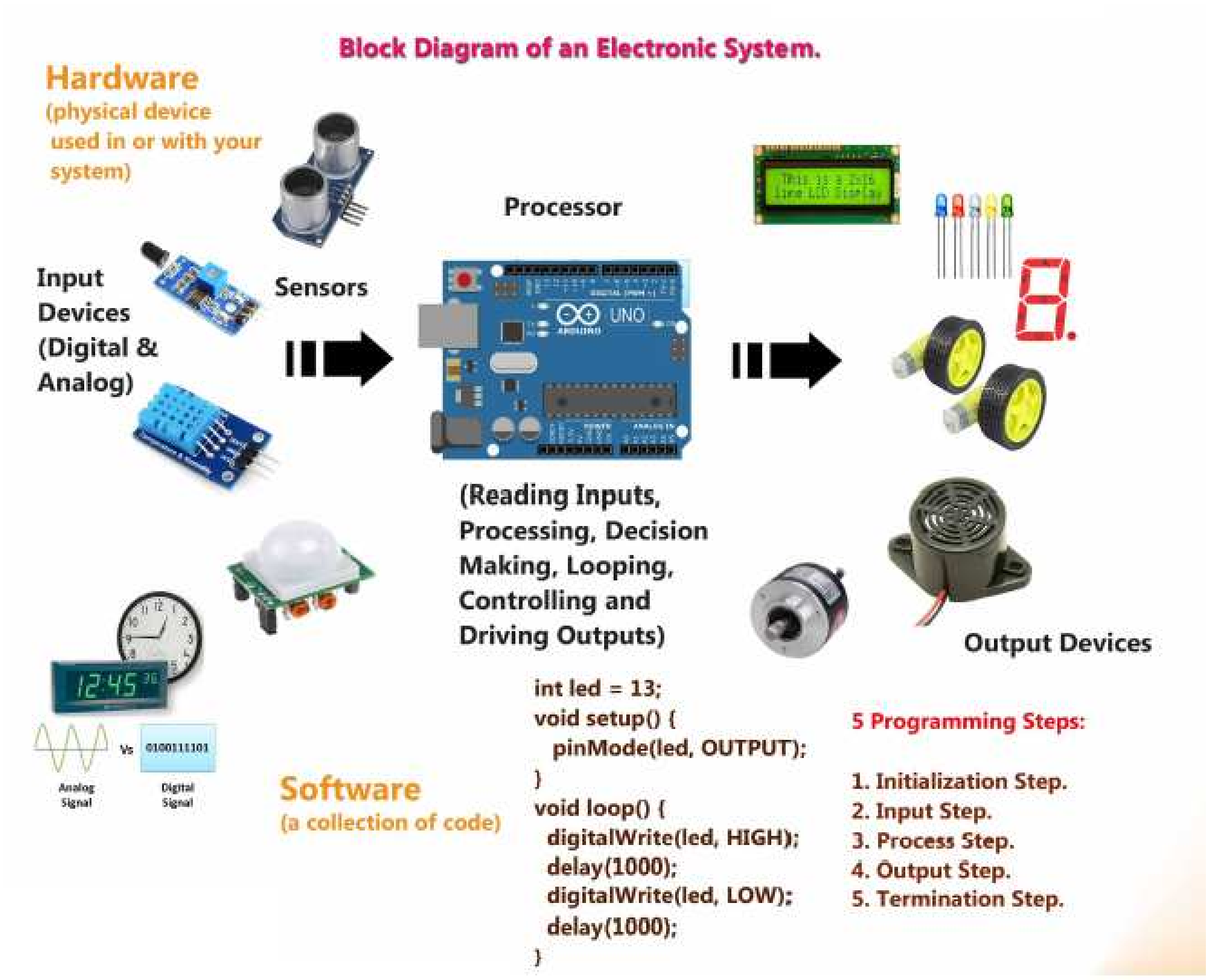


Arduino Program Structure

```
/* Each Arduino sketch must contain
 thefollowing two functions. */

void setup()
  {
     /* this code runs once at the
     beginning of the code execution. */
  }

void loop()
  {
      /* this code runs repeatedly over and
      over as long as the board is powered. */
  }
```



```
int led = 13;
void setup() {
  pinMode(led, OUTPUT);
}
void loop() {
 digitalWrite(led, HIGH);
 delay(1000);
 digitalWrite(led, LOW);
 delay(1000);
}
```

Comments

```
// this is a single line

/* this is
a multiline */
```

A comment in a program is a note or documentation that a programmer writes for himself or other programmers who will look at his code. The computer ignores comments.

Built in Arduino Functions

Pin setup	
pinMode(Pin, INPUT/OUTPUT)	Sets the pin to be either an INPUT or an OUTPUT
pinMode(Pin, INPUT_PULLUP)	Sets the pin to be an input using the Arduino board's built-in pull-up resistor
digitalRead(Pin)	Reads the input at Pin and returns a 1 or 0 (HIGH or LOW)
digitalWrite(Pin, VALUE)	Writes a value of 1 or 0 (HIGH or LOW) to digital pin.
analogRead(Pin)	Reads the analog pin Pin and returns an integer between 0 and 1023
analogWrite(Pin, VALUE)	Emulates analog output VALUE using PWM on Pin (note: only available on pins 3, 5, 6, 9, 10, and 11). VALUE = integer value from 0 to 255.
Time functions	
millis()	Returns the time in milliseconds since the Arduino sketch began running as an unsigned long integer
micros()	Returns the time in microseconds since the Arduino Sketch began running as an unsigned long integer
delay(INTEGER)	Delays program execution for INTEGER milliseconds
delayMicroseconds(INTEGER)	Delays program execution for INTEGER microseconds
Mathematical Functions	
min(i, j)	Returns the lowest of the two values i and j
max(i,j)	Returns the highest of the two values i and j
abs(i)	Returns the absolute value of i
sin(angle)	Returns the sine of an angle in radians
cos(angle)	Returns the cosine of an angle in radians
tan(angle)	Returns the tangent of an angle in radians
sqrt(i)	Returns the square root of i
pow(base, exponent)	Raises the number base to the number exponent (e.g pow (2 , 3) ==8)
constrain(i, minval, maxval)	Contrains the value i between minval and maxval
map(val, fromL, fromH, toL, toH)	Remaps val from one range to another
random(i)	Returns a random long integer smaller than i
random(i, j)	Returns a random long integer between i and j
randomSeed(k)	Uses the value k to seed the random() function
Casting	
(type)variable	Casts the value of variable to a new type
Serial Communication	
Serial.begin(speed)	Start serial communication at a specified speed
Serial.end()	Close serial communication

Serial.print(DATA)	Prints DATA to the serial port. DATA can be characters, strings, integers and floating point numbers
Serial.available()	Return the number of characters available to read in the serial buffer
Serial.read()	Read the first character in the serial buffer (returns -1 if no data is available)

Variables

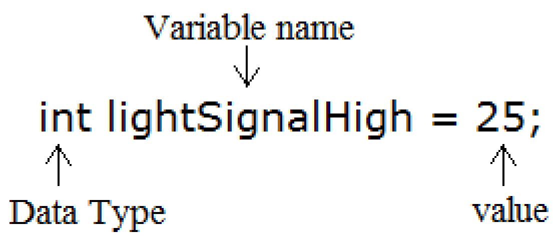
A variable has a name, type, value, and scope.

Declaring a Variable:

int lightSignalHigh;

Assigning a Value to a Variable:

int lightSignalHigh = 25;



Variable scope

```
int ledPin = 12;  // Global variable, can
   //be used throughout the programme.

void setup()
{

int sensorValue = 125; // Local variable,
//can be used in setup function only.

}
void loop()
{

}
```

A variable namemust start with an uppercase or lowercase letter, the underscore (_), or the dollar sign ($). The rest of the name must be made of letters, digits (0–9), the underscore, or thedollar sign. A variable name cannot include spaces. Avariable name should be meaningful and should correctly describe what the variable stores. You should not use Arduino keywords i.e (if, loop, setup, else etc...)as variable or functionnames in your Arduino programs

Data Types

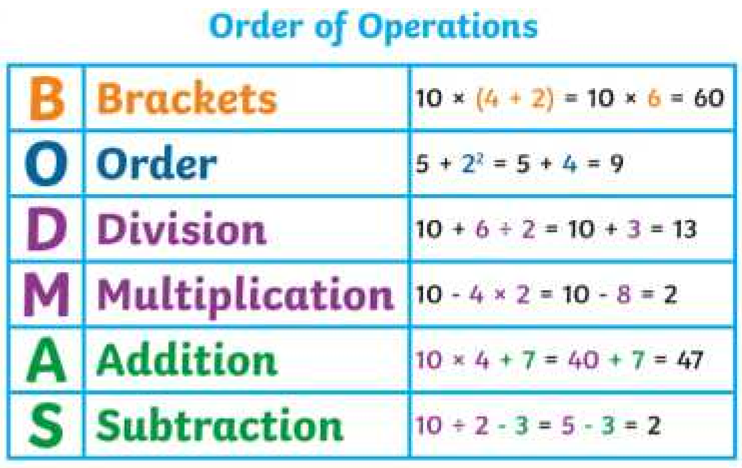
```
void      // nothing is returned
boolean   // 0, 1, false, true
char      // 8 bits: ASCII character
byte      // 8 bits: 0 to 255, unsigned
int       // 16 bits: 32,768 to 32,767, signed
long      // 32 bits: 2,147,483,648 to 2,147,483,647, signed
float     // 32 bits, signed decimal, -3.4028e+38 - 3.4028e+38
```

Operators

Mathematical Operators (Arithmetic Operators)	Comparison Operators	Boolean Operators (Logical Operators)
= // assignment	== // equal to	&& // Boolean AND
+ // addition	!= // not equal to	|| // Boolean OR
- // subtraction	< // less than	! // Boolean NOT
* // multiplication	> // greater than	**Bitwise Operators**
/ // division	<= // less than or equal to	& // bitwise AND
% // modulus		| // bitwise OR
	>= // greater than or equal to	^ // bitwise XOR
		~ // bitwise INVERT
		var << n // shift left by n bits
		var >> n //shift right by n bits

Compound Operators

++	increment
--	decrement
+=	compound addition
-=	compound subtraction
*=	compound multiplication
/=	compound division
&=	compound bitwise AND
|=	compound bitwise OR



Order of Operations

B	Brackets	10 × (4 + 2) = 10 × 6 = 60
O	Order	$5 + 2^2 = 5 + 4 = 9$
D	Division	10 + 6 ÷ 2 = 10 + 3 = 13
M	Multiplication	10 - 4 × 2 = 10 - 8 = 2
A	Addition	10 × 4 + 7 = 40 + 7 = 47
S	Subtraction	10 ÷ 2 - 3 = 5 - 3 = 2

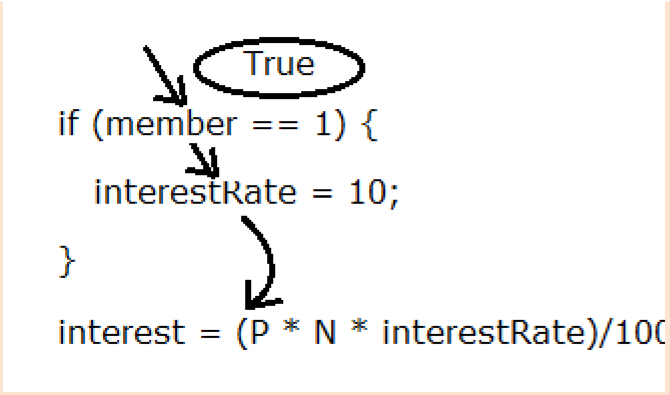
Control statements (Decision Making)

if statement

Syntax	Example1	Example2

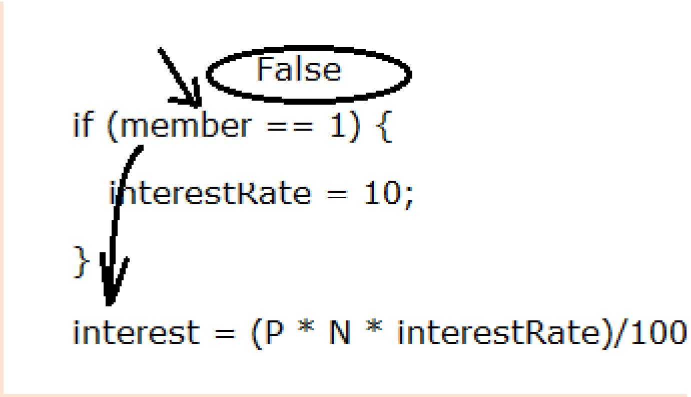
if (condition)
{
Block of statements;
}
	interestRate = 12;
	if (member == 1) {
	interestRate = 10;
	}
		if(lightLevel> 900){
		Serial.print("Light level is: ");
		Serial.println(lightLevel);
		}

Instructions execution sequence for condition True.



```
if (member == 1) {
   interestRate = 10;
}
interest = (P * N * interestRate)/10
```

Instructions execution sequence for condition False.



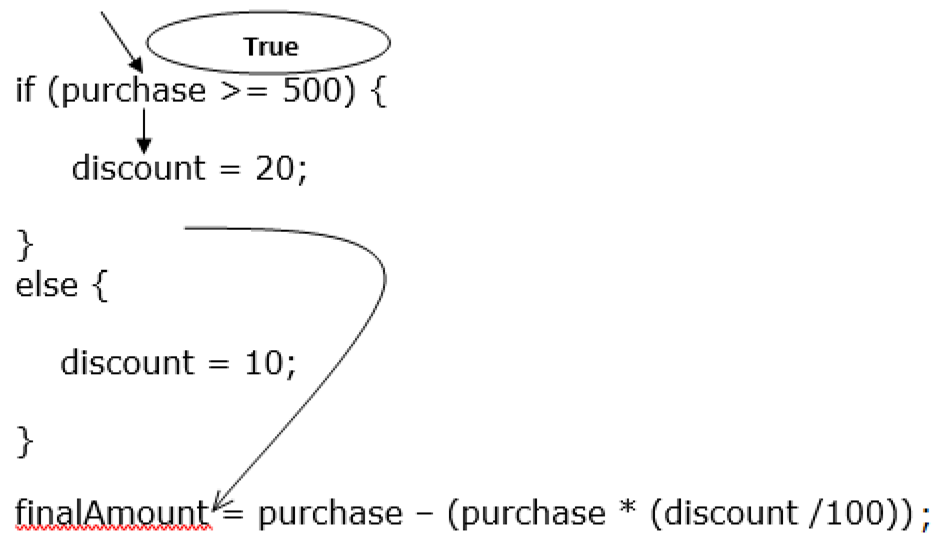
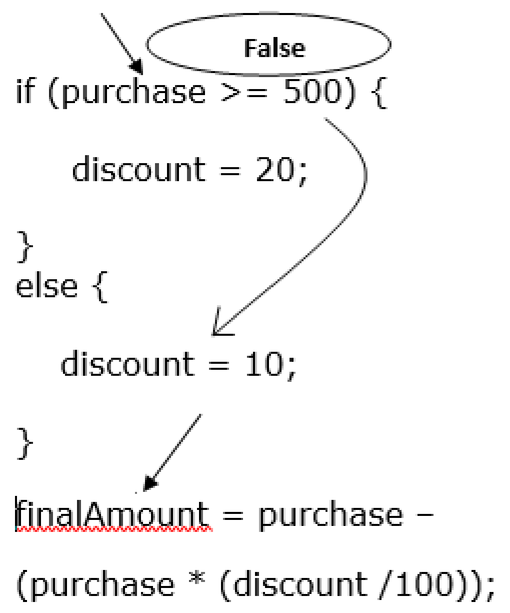
```
if (member == 1) {
   interestRate = 10;
}
interest = (P * N * interestRate)/100
```

if …else statement

Syntax	Example1	Example2

if (condition)
{
Block of statements;
}
else
{
Block of statements;
}
	if (purchase >= 500) {
	discount = 20;
	}
	else {
	discount = 10;
	}
		if (number % 2 == 0) {
		Serial.print(number);
		Serial.println(“: is Even. ”);
		}else{
		Serial.print(number);
		Serial.println(“: is Odd. ”);
		}

Instructions execution sequence for condition True.	Instructions execution sequence for condition False.

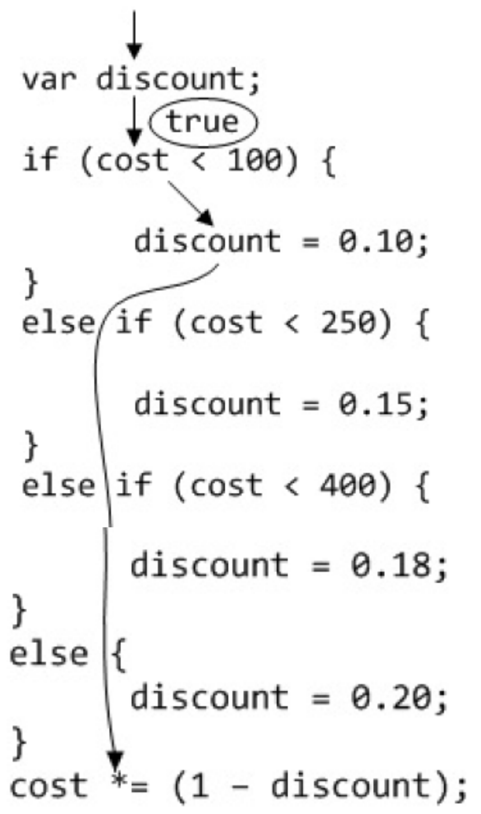
 if (purchase >= 500) {
discount = 20;
}
else {
discount = 10;
}
finalAmount = purchase – (purchase * (discount /100));
	 if (purchase >= 500) {
	discount = 20;
	}
	else {
	discount = 10;
	}
	finalAmount = purchase –
	(purchase * (discount /100));

if … else if …else statement

Syntax	Example1	Example2

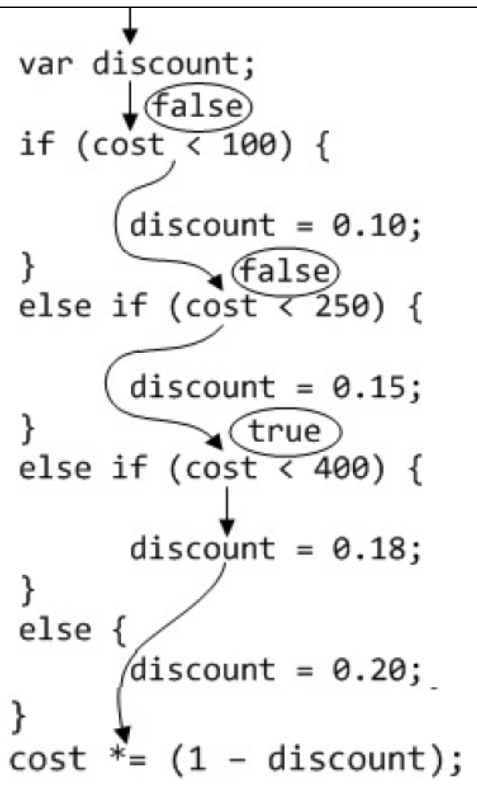
if (condition)
{
Block of statements;
}
else if (condition)
{
Block of statements;
}
else if (condition)
{
Block of statements;
}
….
else {
Block of statements;
}
	`var discount;`
	`if (cost < 100) {`
	`    discount = 0.10;`
	`}`
	`else if (cost < 250) {`
	`    discount = 0.15;`
	`}`
	`else if (cost < 400) {`
	`    discount = 0.18;`
	`}`
	`else {`
	`    discount = 0.20;`
	`}`
	`cost *= (1 - discount);`
		if (score >= 90)
		Serial.println("A");
		else
		if (score >= 80)
		Serial.println("B");
		else
		if (score >= 70)
		Serial.println("C");
		else
		if (score >= 60)
		Serial.println("D");
		else
		Serial.println("F");

Example1: Instructions execution sequence.



```
var discount;
if (cost < 100) {
      discount = 0.10;
}
else if (cost < 250) {
      discount = 0.15;
}
else if (cost < 400) {
      discount = 0.18;
}
else {
      discount = 0.20;
}
cost *= (1 - discount);
```

Example 2: Instructions execution sequence.



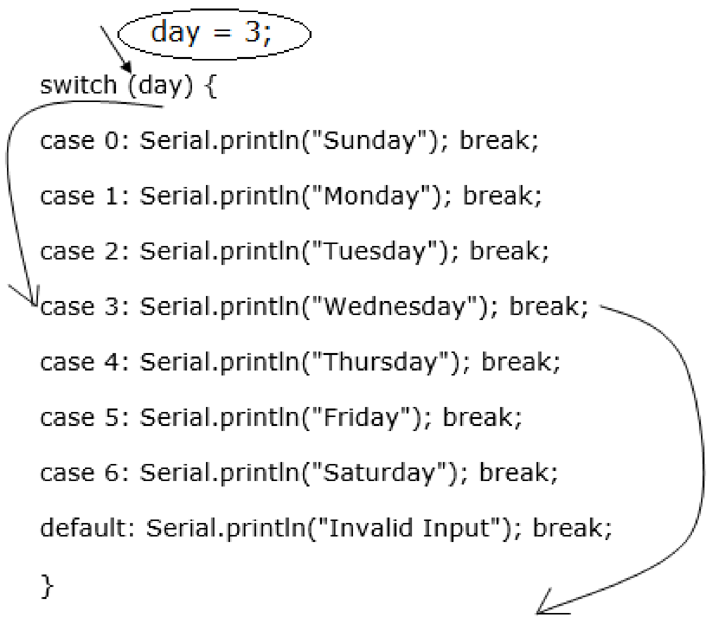
```
var discount;
if (cost < 100) {
      discount = 0.10;
}
else if (cost < 250) {
      discount = 0.15;
}
else if (cost < 400) {
      discount = 0.18;
}
else {
      discount = 0.20;
}
cost *= (1 - discount);
```

Switch Case

Syntex	Example

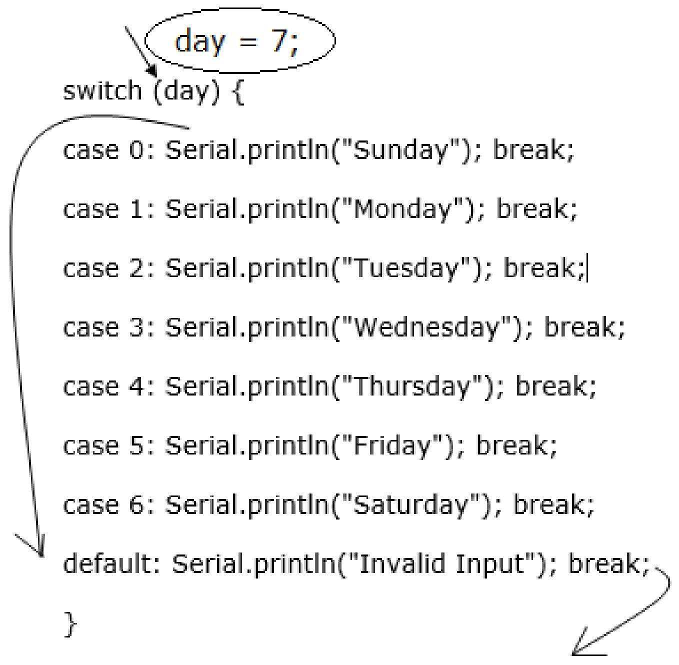
switch (variable) {
case constValue1: statements; break;
case constValue2: statements; break;
…
default: statements; break;
}
	switch (day) {
	case 0: Serial.println("Sunday"); break;
	case 1: Serial.println("Monday"); break;
	case 2: Serial.println("Tuesday"); break;
	case 3: Serial.println("Wednesday"); break;
	case 4: Serial.println("Thursday"); break;
	case 5: Serial.println("Friday"); break;
	case 6: Serial.println("Saturday"); break;
	default: Serial.println("Invalid Input");
	break;
	}

Example1: Instructions execution sequence.



```
day = 3;
switch (day) {
case 0: Serial.println("Sunday"); break;
case 1: Serial.println("Monday"); break;
case 2: Serial.println("Tuesday"); break;
case 3: Serial.println("Wednesday"); break;
case 4: Serial.println("Thursday"); break;
case 5: Serial.println("Friday"); break;
case 6: Serial.println("Saturday"); break;
default: Serial.println("Invalid Input"); break;
}
```

Example2: Instructions execution sequence.



```
day = 7;
switch (day) {
case 0: Serial.println("Sunday"); break;
case 1: Serial.println("Monday"); break;
case 2: Serial.println("Tuesday"); break;
case 3: Serial.println("Wednesday"); break;
case 4: Serial.println("Thursday"); break;
case 5: Serial.println("Friday"); break;
case 6: Serial.println("Saturday"); break;
default: Serial.println("Invalid Input"); break;
}
```

Loops (Repeat a sequence of statements)

While loop

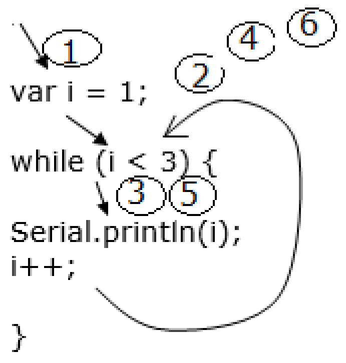
while loop syntax:

```
while (condition) {
    //statements;
}
```

Example:

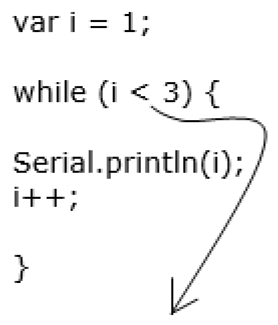
```
var i = 1;
while (i < 3) {
Serial.println(i);
i++;
}
```

Instructions execution sequence for condition True.



```
var i = 1;
while (i < 3) {
Serial.println(i);
i++;
}
```

Instructions execution sequence for condition False.



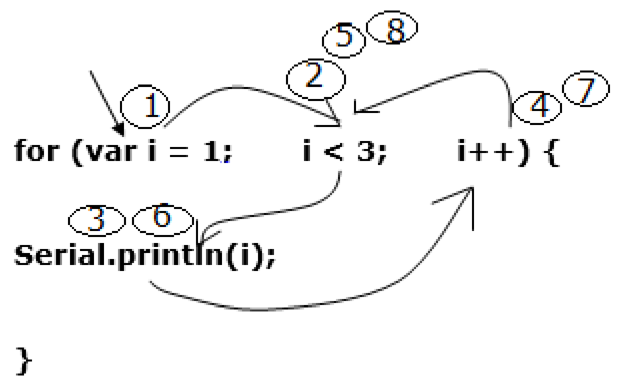
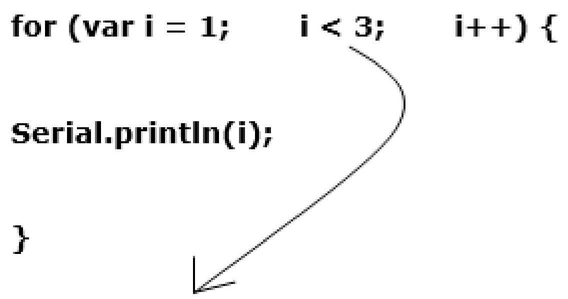
```
var i = 1;
while (i < 3) {
Serial.println(i);
i++;
}
```

For loop

For loop syntax:	Example:

for (initialize; condition; increment or decrement)
{
  // statement block
}
	for (var i = 1; i <= 9; i++) {
	Serial.println(i);
	}

Instructions execution sequence for condition True.	Instructions execution sequence for condition False.

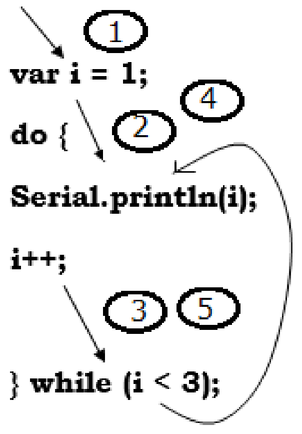
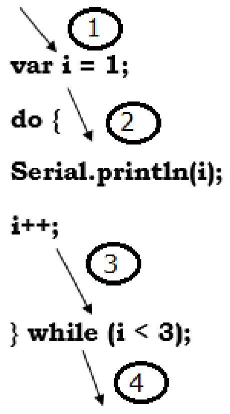
 for (var i = 1; i < 3; i++) {
Serial.println(i);
}
 	
	 for (var i = 1; i < 3; i++) {
	Serial.println(i);
	}

Do-While loop

Do-while loop syntax:	Example:

do{
  //statements;
} while (condition);
	var i = 1;
	do {
	Serial.println(i);
	i++;
	} while (i < 3);

Instructions execution sequence for condition True.	Instructions execution sequence for condition False.

 var i = 1;
do {
Serial.println(i);
i++;
} while (i < 3);
 	
	 var i = 1;
	do {
	Serial.println(i);
	i++;
	} while (i < 3);

Functions (divide programs into parts)

Sum calculation function

```
void setup ()
{

}

void loop ()
{
int result = 0;
result = sumFunction (5,6); // function call
}


int sumFunction (int x, int y) // function declaration
{
int z=0;
z= x+y;
return z; // return the value
}
```

Function Syntax:

```
returnTypex = functionName (argument1, argument2);


   returnType functionName (parameter1, parameter2){
               statements;
                --
                --
         return result;

             }
```

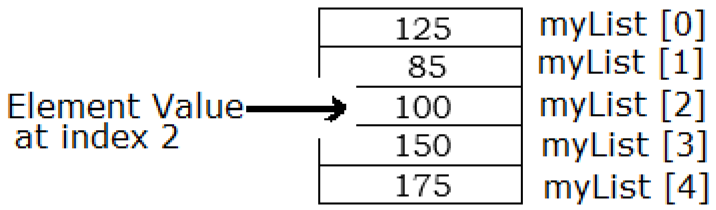
Arrays (a collection of similar variables)

Array declaration:

```
int myList [ 5 ];
```

Array declaration with assigned values:

```
int myList [ ]  =  { 125,85,100,150,175 };
```



Assign a value to the indexed position:

```
myList [3] = 65;
```

To retrieve a value from the array:

```
int x = myList [3];
```

Demonstrating arrays usage:

```
int ledPin = 11;
byte flicker[]  = { 180,30,255,200,10,90,150,60}

void setup (){
   pinMode (ledPin, OUTPUT);
}
void loop() {
  for (int i = 0; i < 7; i++){
  analogWrite(ledPin, flicker[i]);
  delay(200);
   }
}
```