Learning the (other) Code

Learning the (other) Code

- The Arduino Language is a subset of C++
  - C++ and C share many features, so if you only know C, you will still feel at home
- There are Arduino-specific Libraries for extending the Language for specific tasks:
  - Ethernet, WiFi, I2C, SPI, Stepper, Servo, SD, LiquidCrystal, Debounce, FFT,etc

Learning the (other) Code: Online references

- The Arduino Language reference:
  - https://www.arduino.cc/en/Reference/HomePage
- Arduino-specific Libraries:
  - https://www.arduino.cc/en/Reference/Libraries
  - Let’s take a look at the Libraries (if we have wifi)!

C++ Beginners' Trap

- C++ is case sensitive

  RFpower is NOT the same as rfpower is not the same as rfPower

- camelCase is usually used for functions and variables

- PascalCase is usually used for classes

Programming – A Few Terms

- **Comment**
  - Statement that is used to make a program easier to understand, and which is ignored by the computer
    - Comments on a single line start with // in C++

```
// This is a comment in C++
int A = 5;  //This line contains a comment too!
```

    - Multi-line comments are bracketed by /* *and* */

```
/*    This is what a multi-line
      comment in C++ looks like    */
```

Programming – A Few Terms

- **Variable**
  - a memory location paired with an associated symbolic name (an identifier), which contains some known or unknown quantity of information referred to as a value that can be altered during program execution.
    - double myAirspeed = 588.725;
- **Constant**
  - a value that cannot be altered by the program during normal execution
    - const double MyPi = 3.14159;

Programming – A Few Terms

- **Data type**
  - Specifies the <u>type</u> of value for a variable or constant
    - String
      - “This is a string”
    - char(acter)
      - ‘g’
    - int(eger)
      - 3
    - bool(ean)
      - true false 0 (zero) evaluates to false non-zero evaluates to true
    - double or float (identical on the 8 bit boards; 8 vs 4 byte on Due)
      - 3.78943236593

Programming – A Few Terms

- Array
  - An array is a fixed-size sequential collection of elements of the same data type.
  - Ways to declare an array:

    type arrayName[arraySize];

    type arrayName[arraySize] = {value, value, value, value};

    - Example:

      int wattsOut[5] = {10, 20, 30, 40, 50};

    - First element of an array: arrayName[0];

      wattsOut[0] = 10;

      wattsOut[3] = 40;

Programming – A Few Terms

- **Statement**
  - the smallest standalone element of a programming language that expresses some action to be carried out. In C++, every statement ends with a semi-colon
    - X = 1;

Programming – A Few Terms

- **Function**
  - a named section of a program that performs a specific task and returns a value
    - ```
      int X (int y)
      {
          int x = y + 5;
          return x;
      }
      ```
  - “X” would be used in this manner:
    - int A = X(7);
      - Would give A = 12
  - Every function has a type and a name; arguments are optional
    - A function’s “type” represents the data type that it returns when called
      ```

Programming – A Few Terms

- **Procedure**
  - a named section of a program that performs a specific task (such as I/O) but **does not** return a value.
    Example:

```
void calcA (int y)
{
    A = y + 5;
}
```

(where “A” is declared elsewhere in the program)

  - “calcA” would be used in this manner:

```
int A = 0;
calcA(7);
Serial.print(A);
```

    - Would print “12” to the serial port
  - **A procedure’s type is always “void”**
  - Another example:

```
Serial.begin(9600);
```

ARDUINO OFFICIALLY CALLS THIS A “FUNCTION”; SLOPPY SEMANTICS!!

Programming – A Few Terms

- **Class**
  - A type or data structure declared with the keyword class that has data and functions as its members.
  - Instances of a class data type are known as objects.
  - Arduino Class Examples: Serial, String, LiquidCrystal, Stepper, Ethernet, EthernetUDP
  - Creating an instance of a class is called instantiation
  - A class member is accessed using ClassName.member syntax

Programming – A Few Terms

- **Library**
  - a collection of precompiled classes containing functions and procedures that a program can use to give it increased functionality
    - Ethernet.h
    - https://www.arduino.cc/en/Reference/Libraries

Ethernet.h

- Library for Ethernet Shield, Ethernet Shield 2, and Leonardo Ethernet. Contains the classes:

  Ethernet: members begin(), localIP(), maintain()

  IPAddress: member IPAddress()

  Server: members Server, EthernetServer(), begin(), available(), write(), print(), println()

  Client: members Client, EthernetClient(), if(EthernetClient), connected(), connect(), write(), print(), println(), available(), read(), flush(), stop()

  EthernetUdp members begin(), read(), write(), beginPacket(), endPacket(), parsePacket(), available(), stop(), remoteIP(), remotePort()

Programming – A Few Terms

- **Macro**
  - A macro is a preprocessing directive defined by the #define directive:

    #define macro_name macro_body

    - e.g.

      #define X 10

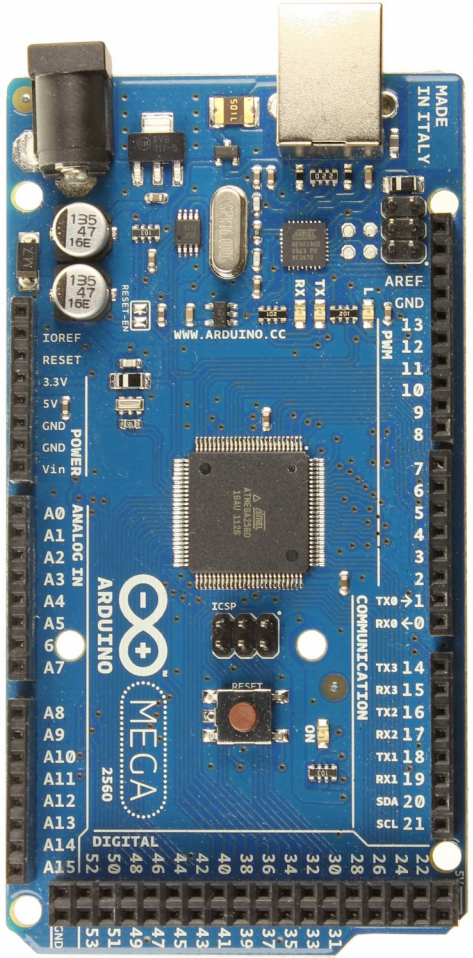
      - Will result in “X” being replaced by “10” everywhere in the code after the #define directive
  - Both #define and #include are preprocessor directives. They change the source code BEFORE compilation. Other commonly used preprocessor directives include: #ifndef and #endif
    - Preprocessor directives are NOT program statements and they do NOT end with a semicolon.

Programming – A Few Terms

- **GPIO pin**
  - generic pin on an integrated circuit or computer board whose behavior—including whether it is an input or output pin—is controllable by the user at run time
    - GPIO.setup(PIN50, GPIO.OUT);
- **Analog pin**
  - Pin that can read (or sometimes write) analog voltages within a defined range with step size determined by the bit size of the analog-to-digital (read) or digital-to-analog (write) converter
    - pinMode(A0, INPUT);

Inputs:

GPIO Pins
Analog Pins
Serial
Ethernet



Outputs:

GPIO Pins --> Relays
Analog Pins --> PWM motor control
Serial --> USB-Serial
Ethernet --> Browser Database

Programming – A Few Terms

- **Operator**
  - A symbol that tells the compiler to perform specific mathematical or logical manipulations
    - Arithmetic
    - Relational
    - Logical
    - Bitwise
    - Assignment
    - Miscellaneous

Operators

- Arithmetic

  + (plus) **-** (minus) ***** (multiply) **/** (divide) **%** (modulus, or remainder)

  ++ (increases integer value by one) **– –** (decreases integer value by one)

- Relational

  == (is equal to) **!=** (is not equal to) **>** (is greater than)

  < (is less than) **>=** (is greater than or equal to)

  <= (is less than or equal to)

- Logical

  && (and) **||** (or) **!** (not)

Operators

- Bitwise

  & (and) **|** (or) **^** (exclusive or) **<<** (shift left) **>>** (shift right)

  ~ (not)

- Assignment

  = (equals)

  += (adds right operand to the left operand and assigns the result to left operand)

  -= (subtracts right operand from the left operand and assigns the result to left operand)

  ***=** (multiplies right operand by the left operand and assigns the result to left operand)

  /= (divides left operand by the right operand and assigns the result to the left operand)

- Miscellaneous

  sizeof() - returns the size (in bytes) of the argument as integer

  (type) cast - converts one data type to another (see next slide)

Cast operator ()

- Cast forces one data type to be converted into another
- Example:

```
double a = 21.123456;
int c;
c = (int) a;
print c;
```

  - Will print the result:

```
21
```

Extra Credit – Bitwise Operator Example

```
/* C Program to demonstrate use of bitwise operators */
#include<stdio.h>
int main()
{
    unsigned char a = 5, b = 9; // a = 5(00000101), b = 9(00001001)
    printf("a = %d, b = %d\n", a, b);
    printf("a&b = %d\n", a&b); // The result is 00000001
    printf("a|b = %d\n", a|b);  // The result is 00001101
    printf("a^b = %d\n", a^b); // The result is 00001100
    printf("~a = %d\n", a = ~a);   // The result is 11111010
    printf("b<<1 = %d\n", b<<1);  // The result is 00010010
    printf("b>>1 = %d\n", b>>1);  // The result is 00000100
    return 0;
}
```

From: https://www.geeksforgeeks.org/interesting-facts-bitwise-operators-c/

Output:

a = 5, b = 9

a&b = 1

a|b = 13

a^b = 12

~a = 250

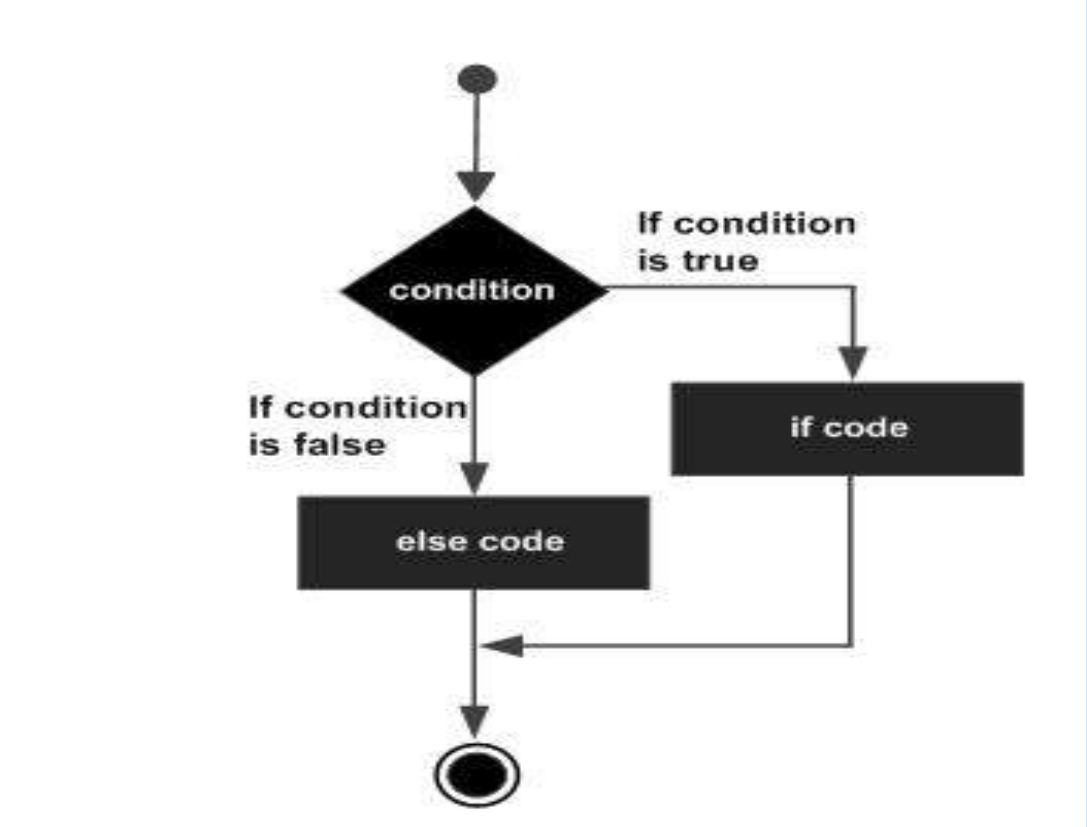
b<<1 = 18

b>>1 = 4

Programming – A Few Terms

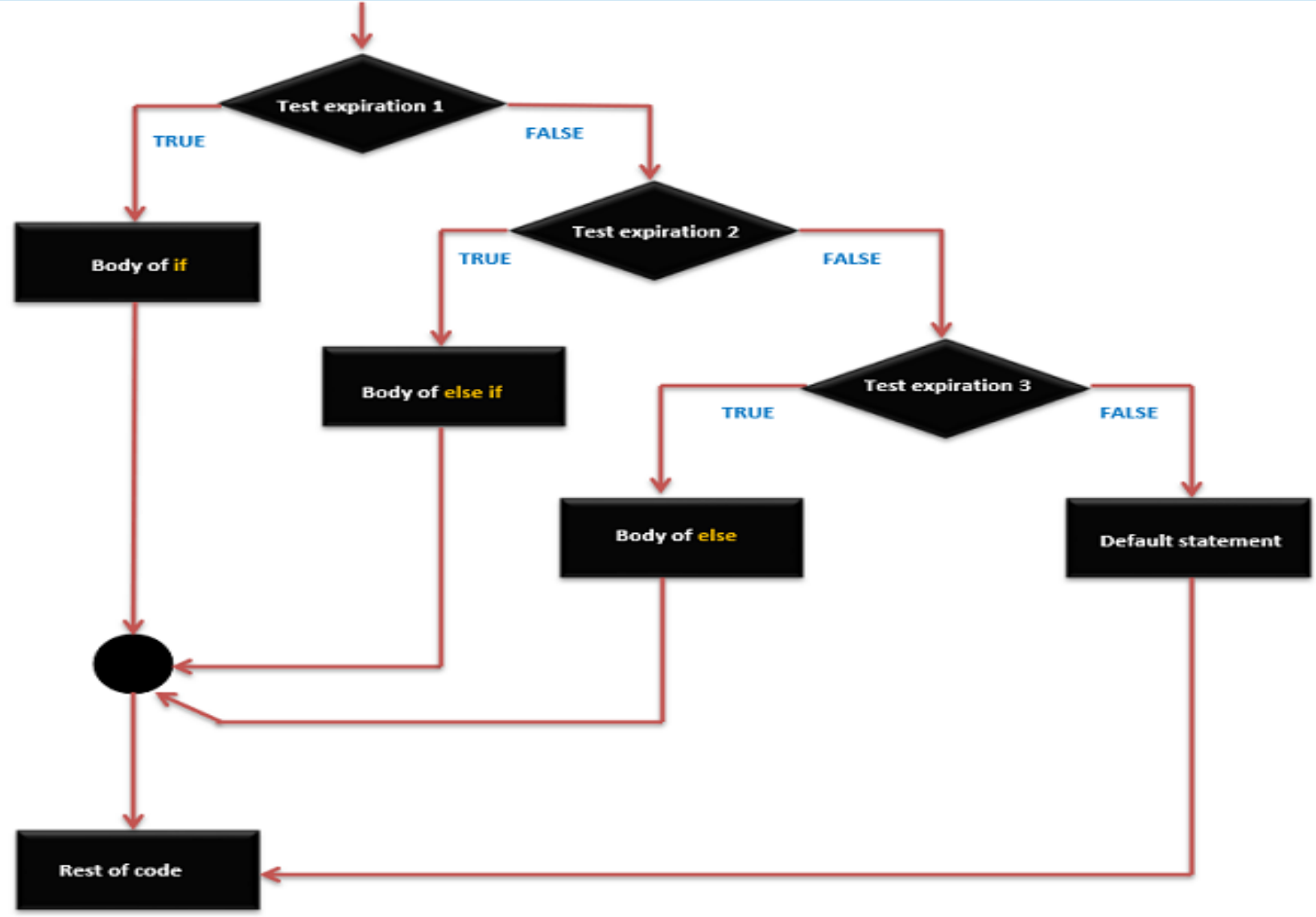
- **Decision Statement**
  - Statement that causes various courses of action to be taken depending on certain conditions (a form of “flow control”)
  - **if, else if, else**
    - if
    - if...else
    - if...else if
    - if...else if...else
    - if...else if...else if
    - if...else if...else if...else etc.
  - **switch**

if/else Statement

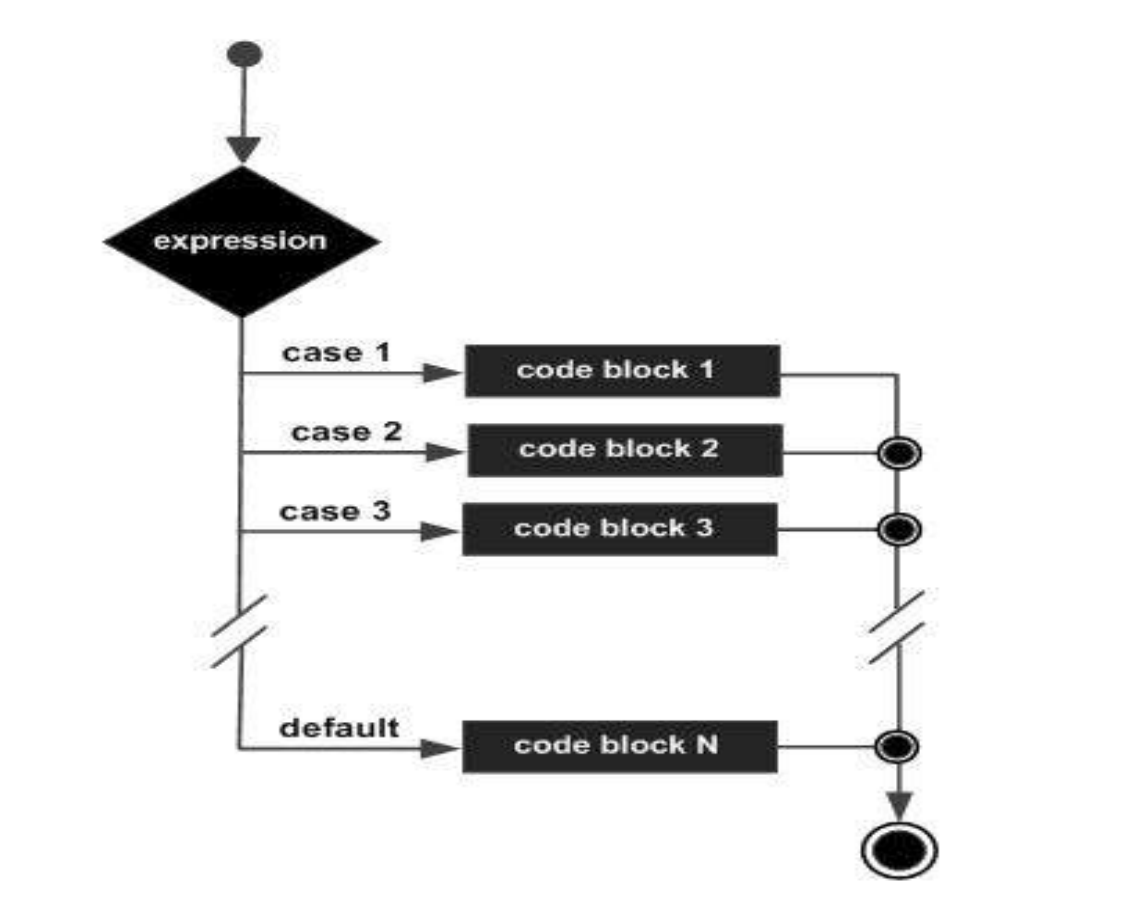


if / else if / else Example

```
if (band == 50)
    {relay50Pin = On;
    relay144Pin = Off;}
else if (band ==144)
    {relay50Pin = Off;
    relay144Pin = On;}
else
    {relay50Pin = Off;
    relay144Pin = Off;}
```


Test expiration 1
TRUE
FALSE
Body of if
Test expiration 2
TRUE
FALSE
Body of else if
Test expiration 3
TRUE
FALSE
Body of else
Default statement
Rest of code

switch : an alternative to if / else if / else if / else if …. / else



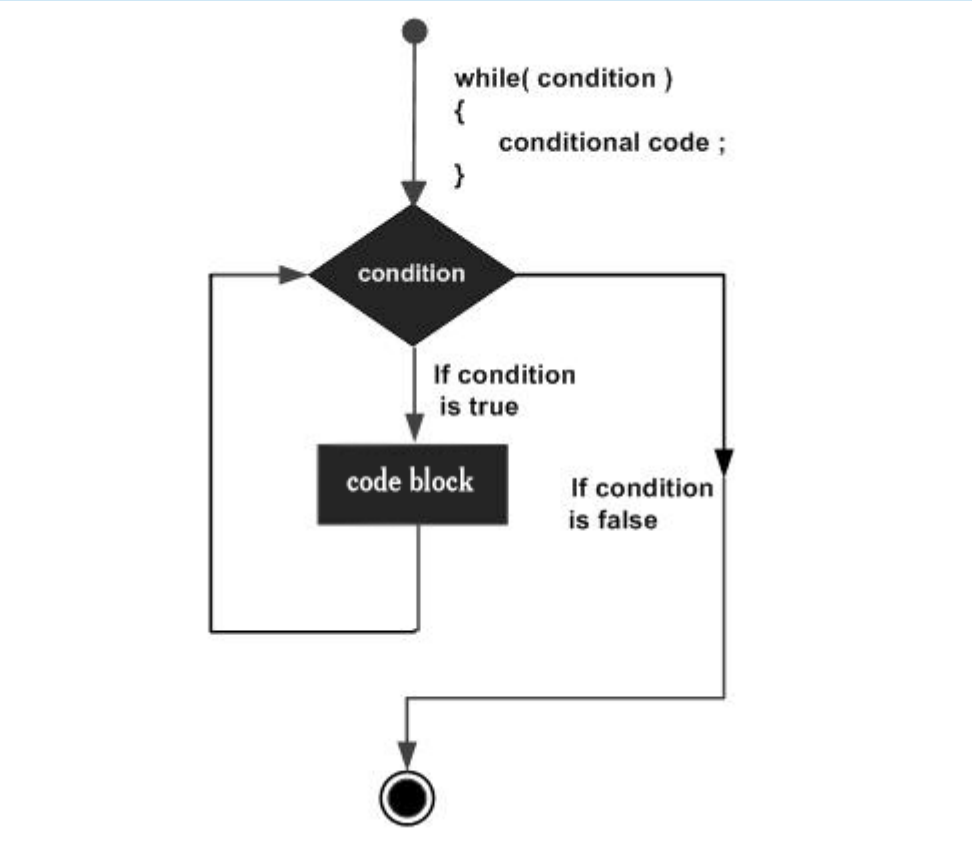
FOR INTEGRAL and ENUMERATED TYPES ONLY
e.g. integers, characters, booleans

Switch Example

```
switch (band){
  case 50: {
      relay50Pin = On;
      relay144Pin = Off;
      break; }
  case 144: {
      relay50Pin = Off;
      relay144Pin = On;
      break; }
  default: {
      relay50Pin = Off;
      relay144Pin = Off; }}
```

Another Form of Flow Control ”while” Loop

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




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

```
var=0;
while(var < 200){
var++;
print var;
}
```

Prints integers from 1 to 200

Another form of flow control the “for” loop

- Used for code that needs to execute repetitively:

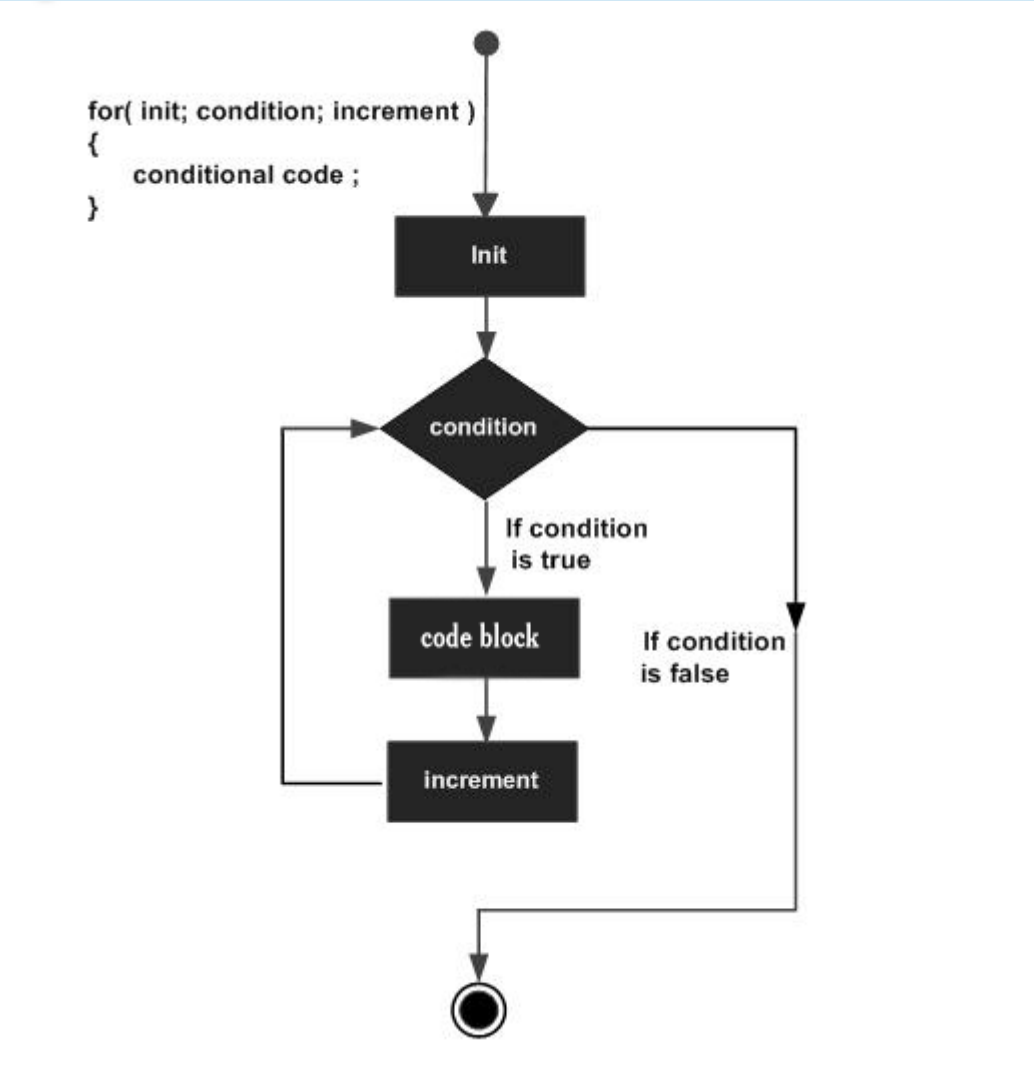
```
for(init; condition; increment) {
    statements; }
```

init step is executed first, and only once

condition evaluated; if true, statements in the body of the loop are executed. If false, body of loop is not executed and control jumps to next statement after the loop

after the body of the loop executes, flow of control jumps to increment statement and increment is performed

condition is again evaluated. When condition is false, loop terminates


for(init; condition; increment)
{
conditional code ;
}
Init
condition
If condition
is true
code block
If condition
is false
increment

“for” loop: example

- Used for code that needs to execute repetitively:

```
for(init; condition; increment) {
    statements; }
```

```
for(i=0; i<4; i++) {
    print i; }
```

Output:
0
1
2
3

Curly braces

{ }

- Are used to group a set of statements; always come in pairs
- Be careful with them; misplaced curly braces are a major source of bugs!

Curly brace madness

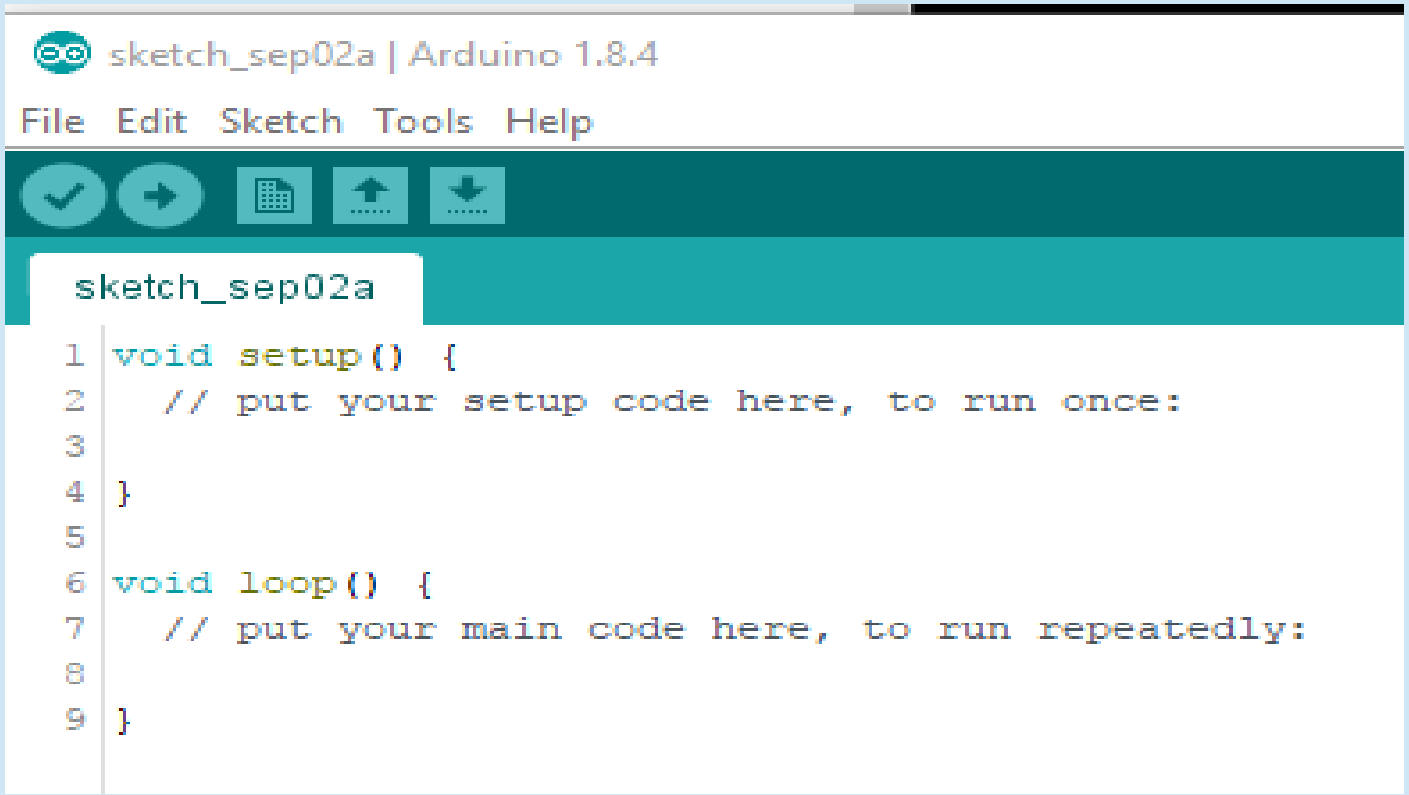
```
if (band == 50)
{relay50Pin = On;
relay144Pin = Off;}
else if (band ==144)
{relay50Pin = Off;
relay144Pin = On;}
else
{relay50Pin = Off;
relay144Pin = Off;}
```

Correct

```
if (band == 50)
{relay50Pin = On;
relay144Pin = Off;}
else if (band ==144)
{relay50Pin = Off;
relay144Pin = On;}
else
{relay50Pin = Off;}
relay144Pin = Off;
```

Incorrect

Programming Steps – Getting Started



Programming Steps - General

1) Include libraries containing classes with external functions (Optional)
2) Define variables and constants (Optional)
3) Setup ()

    Define and initialize GPIO pins / Analog I/O pins

    Define, start, serial port(s), Ethernet port(s)

4) Loop()

    Receive input from ports / GPIO pins / Analog pins

    Parse / process data to extract desired information

    Use information derived from data to perform desired task (e.g. switch GPIO pins) or to send information to client computer

5) From within Loop(), call other functions() as needed (Optional)

A Simple Program
Blink

N1MM_SainSmart_14CH_UNO | Arduino 1.8.4

File Edit Sketch Tools Help

- New Ctrl+N
- Open... Ctrl+O
- Open Recent
- Sketchbook
- Examples
- Close Ctrl+W
- Save Ctrl+S
- Save As... Ctrl+Shift+S
- Page Setup Ctrl+Shift+P
- Print Ctrl+P
- Preferences Ctrl+Comma
- Quit Ctrl+Q

Built-in Examples

- 01.Basics
  - AnalogReadSerial
  - BareMinimum
  - Blink
  - DigitalReadSerial
  - Fade
  - ReadAnalogVoltage
- 02.Digital
- 03.Analog
- 04.Communication
- 05.Control
- 06.Sensors
- 07.Display
- 08.Strings
- 09.USB
- 10.StarterKit_BasicKit
- 11.ArduinoISP

Examples for any board

- Adafruit Circuit Playground
- Bridge
- Esplora
- Ethernet
- Firmata
- GSM
- LiquidCrystal
- Robot Control
- Robot Motor
- SD
- Servo
- SpacebrewYun
- Stepper
- Temboo
- TFT
- WiFi
- RETIRED

Examples for Arduino/Genuino Uno

- EEPROM
- SoftwareSerial
- SPI
- Wire

```
verter bandswitch.

ng data
lete

//define constant pin
const int Pin50 = 2; /
const int Pin144 = 3;
const int Pin222 = 4;
const int Pin432 = 5;
const int Pin902 = 6;
const int Pin1296 = 8;
const int Pin2304 = A5
const int Pin3G = A4;
const int Pin5G = A3;
const int Pin10G = A2;
const int Pin24G = A1;
const int Pin47G = A0;
const int Pin76G = 7;

void setup() {

// define GPIO pins as
pinMode(Pin50,OUTPUT);
pinMode(Pin144,OUTPUT)
pinMode(Pin222,OUTPUT)
pinMode(Pin432,OUTPUT)
pinMode(Pin902,OUTPUT)
pinMode(Pin1296,OUTPUT
pinMode(Pin2304,OUTPUT
pinMode(Pin3G,OUTPUT);
pinMode(Pin5G,OUTPUT);
pinMode(Pin10G,OUTPUT)
pinMode(Pin24G,OUTPUT)
pinMode(Pin47G,OUTPUT)
pinMode(Pin76G,OUTPUT)

//initialize all GPIO
```

```
/*
  Blink

  Turns an LED on for one second, then off for one second, repeatedly.

  Most Arduinos have an on-board LED you can control. On the UNO, MEGA and ZERO
  it is attached to digital pin 13, on MKR1000 on pin 6. LED_BUILTIN is set to
  the correct LED pin independent of which board is used.
  If you want to know what pin the on-board LED is connected to on your Arduino
  model, check the Technical Specs of your board at:
  https://www.arduino.cc/en/Main/Products

  modified 8 May 2014
  by Scott Fitzgerald
  modified 2 Sep 2016
  by Arturo Guadalupi
  modified 8 Sep 2016
  by Colby Newman

  This example code is in the public domain.

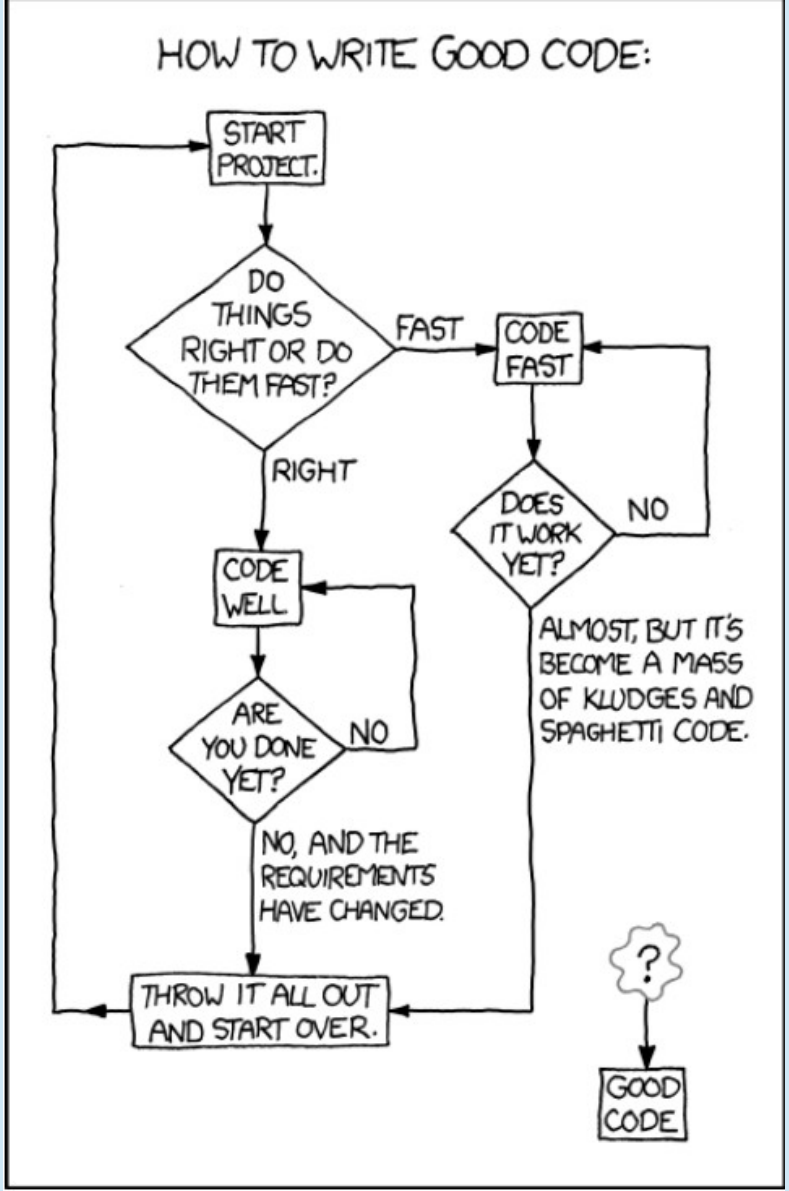
  http://www.arduino.cc/en/Tutorial/Blink
*/

```

```
// the setup function runs once when you press reset or power the board
void setup() {
  // initialize digital pin LED_BUILTIN as an output.
  pinMode(LED_BUILTIN, OUTPUT);
}

// the loop function runs over and over again forever
void loop() {
  digitalWrite(LED_BUILTIN, HIGH);   // turn the LED on (HIGH is the voltage level)
  delay(1000);                       // wait for a second
  digitalWrite(LED_BUILTIN, LOW);    // turn the LED off by making the voltage LOW
  delay(1000);                       // wait for a second
}
```

A Simple Program
Live Demo
Blink


HOW TO WRITE GOOD CODE:
START PROJECT.
DO THINGS RIGHT OR DO THEM FAST?
FAST
CODE FAST
RIGHT
DOES IT WORK YET?
NO
CODE WELL
ALMOST, BUT IT'S BECOME A MASS OF KLUDGES AND SPAGHETTI CODE.
ARE YOU DONE YET?
NO
NO, AND THE REQUIREMENTS HAVE CHANGED.
THROW IT ALL OUT AND START OVER.
?
GOOD CODE