KS2 Cards

CODE
RPI Pico Code layout

Snippet:

```
#Imports here


#Initialise objects here


#Global variables here


#Define functions here


While True:
    #Main code goes here, runs all the time
```

Function:
This a template for laying out your code in a consistent manner, which aids it's development and understanding. **Indents are important**, they place the code inside the function(s).

are comments to give information on what is happening at this point in your code.

CODE
Configure your Pins

Snippet:

Syntax

```
PinName=Pin(#, mode)
```

Parameters

- #: number of the pin to configure
- Mode: Pin.IN or Pin.OUT

Requires

- Pin from machine library

Return

- None

Example

```
LED = Pin(25, Pin.OUT)
```

Function:
This command is used to set the operation of individual pins.

CODE
Operate a Digital Output

Snippet:

Syntax

```
PinName.value(val)
```

Parameters

- pinName: the name set in the pin setup
- val: HIGH / LOW or 0 / 1

Requires

- Pin from machine library

Return

- None

Example

```
LED.value(1)
```

Function:
If the pin has been configured as an **OUTPUT** with the pin method, its voltage will be set to the corresponding value: 3.3V for **HIGH**, 0V (ground) for **LOW**.

CODE
Reading a Digital input

Snippet:

Syntax

```
Var = pinName.value()
```

Parameters

- pinName: the name of the digital pin you want to read

Requires

- Pin from Machine library

Return

- Low / 0 or high / 1

Example

```
Button = switch.value()
```

Function:
Reads the value from a specified digital pin, either **HIGH (1)** or **LOW(0)**.

CODE
Operate an Analog Output

Snippet:

Syntax

```
PinName.duty_u16(value)
```

Parameters

- pinName: the pin to write to
- value: the duty cycle: between 0 (always off) and 65534 (always on)

Requires

- PWM from machine library

Return

- None

Example

```
LED.duty_u16(500)
```

Function:
Writes an analog value (**PWM wave)** to a pin. Can be used to light a LED at varying brightnesses or drive a motor at various speeds. After a call to duty_u16(), the pin will generate a steady square wave of the specified duty cycle until the next command on the same pin.

CODE
Reading an **Analog** pin

Snippet:

Syntax

```
Var = pinName.read_u16()
```

Parameters

- var: variable to store the value into
- pinName: name of analog pin

Requires

- ADC from machine library

Return

- int (0 to 65535), (0V to 3.3V)

Example

```
val = analog.read_u16()
```

Function:
Reads the value on a specified analog pin. The RPI Pico 3 channels, 12-bit analog to digital converter. This will map input voltages between 0 and 3.3V into integer values between 0 and 65535.

CODE
How to delay for a set time

Snippet:

Syntax

```
sleep(ms)
```

Parameters

- Time: in seconds as a decimal number

Requires

- Sleep from utime library

Return

- None

Example

```
Sleep(0.5)
```

Function:
Pauses the program for the amount of time in seconds as a decimal number.

The range of sleep is 0.0 to 10.0.

CODE
Looping for a **While**

Snippet:

Syntax

```
While expression:
  # statement(s)
```

Parameters

- expression: a statement that evaluates to true or false

Return

- None

Example

```
var = 0
while var < 200:
  #do something repetitive 200 times
  var += 1
```

Function:
while loops will loop continuously, and infinitely, until the expression becomes false. Something must change the tested variable, or the while loop will never exit. This could be in your code, such as an incremented variable, or an external condition, such as testing a sensor.

KS2 Cards

CODE

Looping **For** a number of times

Snippet:

Syntax

```
for var in range(start, end, step):
    #statement(s)
```

Parameters

- Var: variable name
- Start: first number
- End: last number + 1
- Step: value to adjust loop by - optional

Return

- None

Example

```
for x in range(1, 10):
    Print(x)
```

Function:

The for statement is used to repeat a block of statements inside the indent. It repeated loops until the it reaches the end value.
You can add a step value if you what to alter the normal increment of 1: (start, end, step). Step can be either positive or negative to count down.

CODE

If decision making

Snippet:

Syntax

```
if someVariable > 50:
    #do something here
```

Parameters

- Comparisons: ==, !=, >, <, <=, >=

Returns

- None

Example

```
a = 33
b = 200
if b > a:
  print("b is greater than a")
```

Function:

if, is used in conjunction with a comparison to tests whether a certain condition has been reached, such as an input being above a certain number. if the comparison is true, the statements inside the brackets are run. If not, the program skips over the code.

CODE

If ... Else decision making

Snippet:

Syntax

```
if pinFiveInput < 500:
    #action A
Else:
    #action B
```

Parameters

- Comparisons: ==, !=, >, <, <=, >=

Returns

- None

Example

```
a = 200
b = 33
if b > a:
  print("b is greater than a")
else:
  print("b is not greater than a")
```

Function:

if/else allows greater control over the flow of code than the basic if statement, by allowing multiple tests to be grouped together. For example, an analog input could be tested and one action taken if the input was less than 500, and another action taken if the input was 500 or greater.

CODE

Basic maths

Snippet:

Arithmetic Operators

The basic maths operations are:

- + addition
- -Subtraction
- * multiplication
- / division
- % modulo

Examples

```
val1 = val + 2
val1 = val - 2
val1 = val * 2
val1 = val / 2
```

Function:

The maths operations follow standard maths procedures, brackets can be used to ensure the correct order of calculation (precedence)

CODE

Create a variable

Snippet:

Syntax

```
name = value
```

Parameters

- name: meaningful name for you variable
- Value: the value you wish to store in the variable

Return

- None

Example

```
delay = 0.1
```

Function:

This is the method use to set up variables, they need to be initialised at the start of your code - see the RPI Pico code layout card.

CODE

Choosing a **Random** number

Snippet:

Syntax

```
Var = randint(min, max)
```

Parameters

- min: lowest number
- max: highest number

Requires

- randint from random library

Returns

- a random number between min and max

Example

```
#print a random number from 1 to 6
randNumber = randint(1, 6)
```

Function:

The random function produces pseudo random numbers, in the range min to max.

CODE

Playing musical **Tone**s

Snippet:

Syntax

```
pinName.duty(1000)
pinName.freq(freq)
```

Parameters

- pinName: pin for PWM to generate tone
- frequency: frequency of tone in hertz

Requires

- PWM from machine library

Returns

- None

Example

```
buzzer = PWM(Pin(15))
buzzer.freq(500)
buzzer.duty_u16(1000)
sleep(1)
buzzer.duty_u16(0)
```

Function:

Generates a square wave of the specified frequency on a pin. The tone continues until a duty(0) command is used.

The pin can be connected to a piezo buzzer or other speaker to play tones.

CODE

Set **Servo** motor position

Snippet:

Syntax

```
pinName.duty(50) #set at start
pinName.duty_u16(angle * 44 + 1000)
```

Parameters

- angle: angle 0 to 180

Requires

- PWM from machine library

Example

```
servo = PWM(Pin(15))
servo.freq(50)

angle = 0
servo.duty_u16(angle * 44 + 1000)
```

Function:

Standard servos allow the shaft to be positioned at various angles, usually between 0 and 180 degrees.

Continuous rotation servos allow the rotation of the shaft to be set to various speeds.