พื้นฐานไพธอนเพื่อการใช้บอร์ดอาร์ดูอิโน่

BASIC PYTHON FOR ARDUINO

โดย ร้อยโท ดร.ชัชรินทร์ เลิศยศบดินทร์

Basic Python for Arduino

พื้นฐานไพธอนเพื่อการใช้บอร์ดอาร์ดูอิโน่

โดย

ร้อยโท ดร.ชัชรินทร์ เลิศยศบดินทร์

Basic Python for Arduino
พื้นฐานไพธอนเพื่อการใช้บอร์ดอาร์ดูอิโน่

ร้อยโท ดร.ชัชรินทร์ เลิศยศบดินทร์
กองวิชาวิทยาศาสตร์ โรงเรียนเตรียมทหาร สถาบันวิชาการป้องกันประเทศ

พิมพ์ครั้งที่ 1 พ.ศ. 2566 จำนวน 100 เล่ม

ข้อมูลทางบรรณานุกรมของหอสมุดแห่งชาติ

National Library of Thailand Cataloging in Publication data

ชัชรินทร์ เลิศยศบดินทร์, ร.ท.
Basic Python for Arduino พื้นฐานไพธอนเพื่อการใช้บอร์ดอาร์ดูอิโน่. – นครนายก : โรงเรียนเตรียมทหาร สถาบันวิชาการป้องกันประเทศ, 2566.
37 หน้า.

1. ไพธอน (ภาษาคอมพิวเตอร์). 2. อาดูยโน (เครื่องควบคุมโปรแกรม). I. ชื่อเรื่อง.

005.133
ISBN 978-616-604-509-3

จัดพิมพ์โดย

ร้อยโท ดร.ชัชรินทร์ เลิศยศบดินทร์ จัดพิมพ์แบบ Print on Demand ที่
โรงเรียนเตรียมทหาร สถาบันวิชาการป้องกันประเทศ
เลขที่ 9 หมู่ 10 ตำบลศรีกะอาง อำเภอบ้านนา จังหวัดนครนายก รหัสไปรษณีย์ 26110

สารบัญ

คำนำ

หนังสือเล่มนี้มีวัตถุประสงค์เพื่อเป็นเอกสารประกอบการฝึกอบรมในหลักสูตร “การโค้ดดิ้งร่วมกับไมโครคอนโทรลเลอร์สำหรับการพัฒนาหุ่นยนต์และระบบอัตโนมัติ” ของสถาบันพัฒนาบุคลากรสาขาเทคโนโลยีการผลิตอัตโนมัติและหุ่นยนต์ (MARA) กรมพัฒนาฝีมือแรงงาน กระทรวงแรงงาน

ผู้แต่งในฐานะวิทยากรประจำหลักสูตร ได้เรียบเรียงเนื้อหาเกี่ยวกับการเขียนโค้ดโปรแกรมภาษาไพธอน (Python) เพื่อควบคุมบอร์ดไมโครคอนโทรเลอร์อาดูอิโน่ (Arduino) ซึ่งจะเป็นพื้นฐานสำคัญในการพัฒนาหุ่นยนต์และระบบอัตโนมัติ

ในหนังสือเล่มนี้ผู้อ่านจะได้ศึกษาตัวอย่างโค้ดที่จำเป็นในการทำความเข้าใจไวยากรณ์ที่สำคัญของการเขียนโปรแกรมภาษาไพธอน ตลอดจนการสร้างซอฟท์แวร์ส่วนต่อประสานกับผู้ใช้ (GUI) เพื่อควบคุมหุ่นยนต์ประเภทแขนกล

ผู้แต่งหวังเป็นอย่างยิ่งว่าหนังสือเล่มนี้จะเป็นประโยชน์ไม่มากก็น้อยต่อผู้อ่านซึ่งมีความมุ่งหมายที่จะศึกษาวิธีการเขียนโปรแกรมควบคุมหุ่นยนต์และอัตโนมัติด้วยภาษาไพธอน

ร้อยโท ดร. ชัชรินทร์ เลิศยศบดินทร์

ผู้แต่ง

บทที่ 1 การติดตั้ง Python

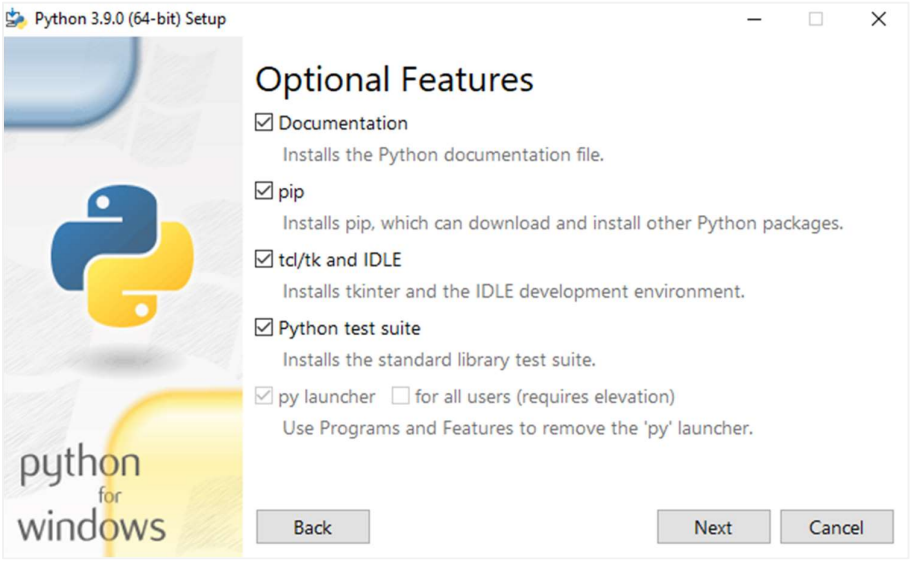
Python เป็นภาษาเขียนโปรแกรมระดับสูงที่ใช้กันอย่างกว้างขวางในการเขียนโปรแกรมสำหรับวัตถุประสงค์ทั่วไป ภาษา Python นั้นสร้างโดย Guido Van Rossum และถูกเผยแพร่ครั้งแรกในปี ค.ศ. 1991 โดย Python นั้นเป็นภาษาแบบ interpreter ที่ถูกออกแบบโดยยึดหลักการที่จะทำให้โค้ดอ่านได้ง่าย และมีโครงสร้างของภาษาที่สามารถเข้าใจได้ง่าย

ก่อนที่จะเริ่มต้นการเขียนโปรแกรมด้วยภาษา Python นั้นจะต้องเริ่มด้วยการติดตั้งเครื่องมือในการพัฒนาโปรแกรม ซึ่งซอฟต์แวร์เหล่านี้เป็นซอฟต์แวร์ฟรีจาก Python โดยคุณต้องไปที่ลิงก์ https://www.python.org/ftp/python/3.9.0/python-3.9.0-amd64.exe

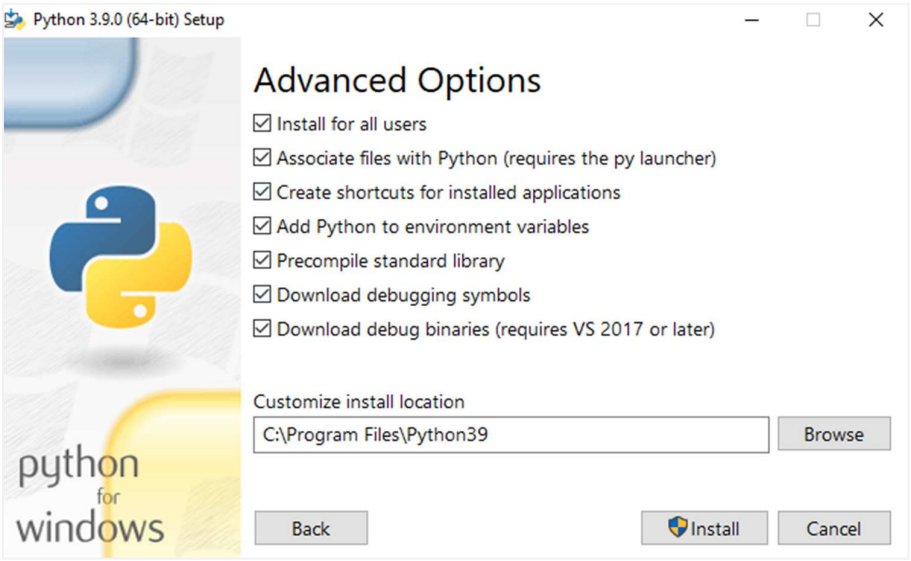
หลังจากที่คุณได้ทำการดาวน์โหลด ไฟล์ตัวติดตั้ง python-3.9.0-amd64.exe เรียบร้อยแล้ว ต่อไปจะเป็นการติดตั้ง Python ลงบนคอมพิวเตอร์ของคุณ โดยให้คุณเลือกช่อง “Add Python 3.9 to PATH” แล้วกดเลือกเมนู “Customize installation”

เมื่อกดเลือกเมนู “Customize installation”แล้วจะเข้าสู่หน้าต่าง “Optional Features” ให้คุณเลือก option ในรายการทั้งหมด แล้วกด “Next” จากนั้นจะเข้าสู่หน้าต่าง “Advanced Options” ให้คุณเลือก option ในรายการทั้งหมด แล้วกด “Install”

รูปแสดงหน้าต่างติดตั้ง Python

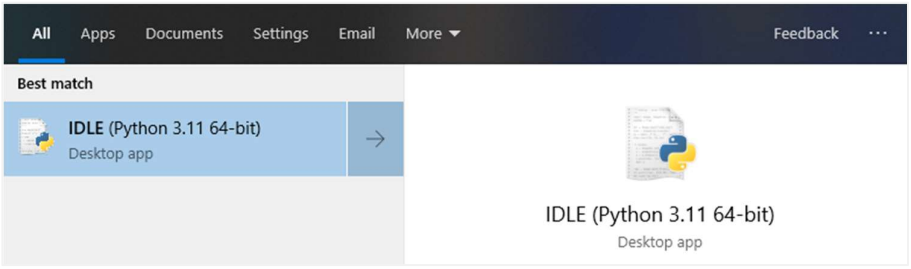


รูปแสดงหน้าต่าง Optional Features จากการเลือกเมนู“Customize installation”

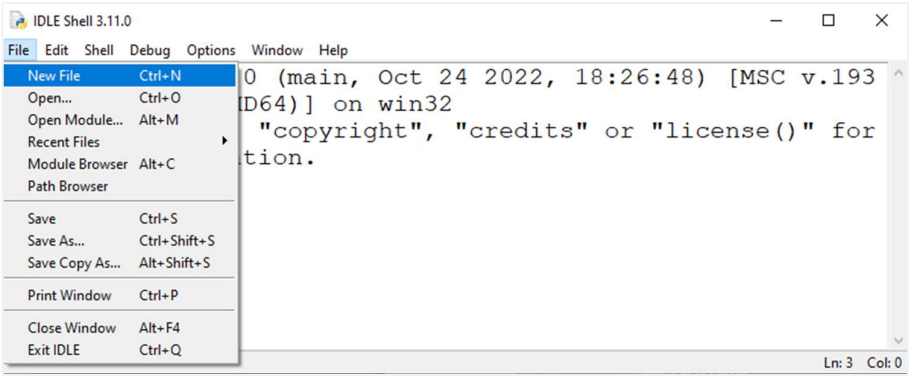


รูปแสดงหน้าต่าง Advanced Options

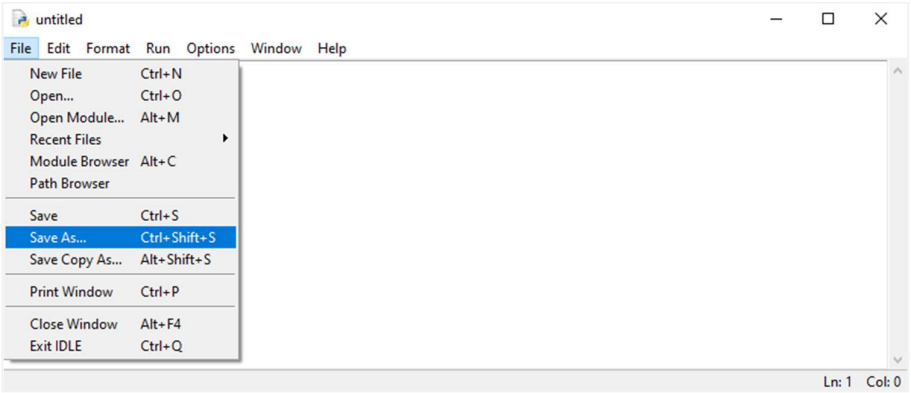
เมื่อคุณติดตั้ง Python เรียบร้อยแล้วให้ใช้ IDLE เพื่อเปิดเครื่องมือเขียนโค้ด Python ทั้งนี้ให้ท่านเลือกเมนู “File” > “New File” บนหน้าต่าง IDLE Shell เพื่อเปิด Text Editor หน้าต่างใหม่ หลังจากนั้นให้ท่านเลือกเมนู “File” > “Save as...” บนหน้าต่าง Text Editor แล้วเลือกพื้นที่เก็บไฟล์พร้อมตั้งชื่อไฟล์ใหม่ซึ่งลงท้ายด้าย .py ยกตัวอย่างเช่น test.py



รูปแสดงไอคอน IDLE



รูปแสดง IDLE Shell



รูปแสดง IDLE Text Editor

รูปแสดง Text Editor ของไฟล์ test.py

บทที่ 2 โครงสร้างของภาษา Python

ในบทนี้ คุณจะได้เรียนรู้และทำความเข้าใจเกี่ยวกับโครงสร้างของภาษา Python ซึ่งสิ่งเหล่านี้ถูกกำหนดเพื่อเป็นรูปแบบที่คุณจะเขียนโค้ดของคุณเพื่อให้เข้าใจโดยตัวแปลภาษาหรือคอมไพเลอร์

ตอนที่ 1 Simple Python program

เพื่อเริ่มต้นการเรียนรู้ภาษา Python มาดูตัวอย่างของโปรแกรมอย่างง่าย โดยเป็นโปรแกรมที่แสดงข้อความทักทายทางหน้าจอ มาเริ่มเขียนโปรแกรมแรกในภาษา Python ของคุณโดยให้คัดลอกโปรแกรมข้างล่างแล้วนำไปรันใน IDE

```
print ('Welcome to Python.')
```

ผลลัพธ์

```
Welcome to Python.
```

ตอนที่ 2 Comment

คอมเมนท์ในภาษา Python นั้นเริ่มต้นด้วยเครื่องหมาย # ตามด้วยคำอธิบาย ซึ่งโดยทั่วไปแล้วคอมเมนท์มักจะใช้สำหรับอธิบายซอสโค้ดที่เราเขียนขึ้นและมันไม่มีผลต่อการทำงานของโปรแกรม โดยชุดคำสั่งต่อไปนี้เป็นตัวอย่างการคอมเมนท์ในภาษา Python

```
# My first Python program
print ('Hello Python.') # Inline comment
```

ผลลัพธ์

```
Hello Python.
```

ตอนที่ 3 Statement

Statement คือคำสั่งการทำงานของโปรแกรม โดยแต่ละคำสั่งในภาษา Python นั้นจะแบ่งแยกด้วยการขึ้นบรรทัดใหม่ หรือใช้เครื่องหมายเซมิโคลอน (;) ในการแบ่งแยกคำสั่งหลายคำสั่งในบรรทัดเดียวกันในตอนนี้ให้คุณคัดลอกโปรแกรมข้างล่างแล้วนำไปรันใน IDE

```
print ('Sa-Wad-Dee')
print ('Welcome to Python.'); print ('Do you love it?')
```

ผลลัพธ์

```
Sa-Wad-Dee
Welcome to Python.
Do you love it?
```

ตอนที่ 4 Indentation and white space

ในภาษา Python นั้นใช้ White space และ Tab สำหรับกำหนดบล็อคของโปรแกรม เช่น คำสั่ง If, Else, For หรือการประกาศฟังก์ชัน(def) ซึ่งคำสั่งเหล่านี้เป็นคำสั่งแบบบล็อค โดยจำนวนช่องว่างที่ใช้นั้นต้องเท่ากัน มาดูตัวอย่างของบล็อคคำสั่งในภาษา Python

```
n = 5
if (n > 0):
        print ('x is positive number')
else:
        print ('x is not positive number')
```

ผลลัพธ์

```
x is positive number
```

ตอนที่ 5 Expressions

Expression คือการทำงานร่วมกันระหว่างตัวแปร และตัวดำเนินการ โดยในภาษา Python นั้นมี Expression อยู่สองแบบ

1) แบบแรกคือ Boolean expression เป็นการกระทำกันระหว่างตัวแปรและตัวดำเนินการเปรียบเทียบค่าหรือตัวดำเนินการตรรกศาสตร์ และจะได้ผลลัพธ์เป็น Boolean
2) แบบที่สองคือ Mathematic expression เป็นการกระทำกันระหว่างตัวแปรและตัวดำเนินการคณิตศาสตร์ และจะได้รับค่าใหม่เป็นตัวเลขหรือค่าที่ไม่ใช่ Boolean

ในตอนนี้ให้คุณคัดลอกโปรแกรมข้างล่างแล้วนำไปรันใน IDE

```
a = 4
b = 5

# Boolean expressions
print(a == 4)
print(a == 5)
print(a == 4 and b == 5)
print(a == 4 or b == 8)

# Non-boolean expressions
print(a + b)
print(a + 2)
print(a * b)
print("Python " + "Language")
```

ผลลัพธ์

```
True
False
True
True
9
6
20
Python Language
```

ตอนที่ 6 Keywords

Keyword เป็นคำที่ถูกสงวนไว้ในการเขียนโปรแกรมภาษา Python เราไม่สามารถใช้คำเหล่านี้ในการตั้งชื่อตัวแปร, ชื่อฟังก์ชัน, ชื่อคลาส หรือตัวระบุชื่อใด ๆ ที่กำหนดขึ้นโดยโปรแกรมเมอร์ ตารางต่อไปนี้เป็นรายการของ Keyword ในภาษา Python

False	None	True	and
as	assert	break	class
continue	def	del	elif
else	except	finally	for
from	global	if	import
in	is	lambda	nonlocal
not	or	pass	raise
return	try	while	with
yield			

ในบทนี้ คุณได้เรียนรู้เกี่ยวกับโครงสร้างของภาษา Python แบบเบื้องต้นไปแล้ว และสิ่งเหล่านี้จะใช้เป็นข้อกำหนด หรือกฎเกณฑ์ในการเขียนโปรแกรมทุก ๆ โปรแกรมที่คุณเขียน

บทที่ 3 ตัวแปรและประเภทข้อมูล

ในบทนี้ คุณจะได้เรียนรู้เกี่ยวกับตัวแปรและประเภทข้อมูลในภาษา Python เราจะพูดถึงการประกาศตัวแปรและการนำตัวแปรไปใช้งานในโปรแกรม

ตอนที่ 1 ตัวแปรประเภท Numbers

ในภาษา Python นั้นสนับสนุนข้อมูลแบบตัวเลขโดยแบ่งออกเป็นประเภท Integer, Float, Decimal และ Complex ซึ่งประเภทของตัวแปรที่มักถูกใช้บ่อย ๆ ในการเขียนโปรแกรมควบคุมหุ่นยนต์ คือตัวแปรประเภท Integer ซึ่งเป็นการเก็บข้อมูลแบบจำนวนเต็ม รวมถึง Float ซึ่งเป็นข้อมูลแบบจำนวนจริง ในตอนนี้ให้คุณคัดลอกโปรแกรมข้างล่างแล้วนำไปรันใน IDE

```
a = 1
b = 2
c = 3.21
d = a + b
e = a / b
f = a - b

print ('a = %d' % a)
print ('b = %d' % b)
print ('c = %f' % c)
print ('d = %f' % d)
print ('e = %f' % e)
print ('f = %d' % f)
```

ผลลัพธ์

```
a = 1
b = 2
c = 3.210000
d = 3.000000
e = 0.500000
f = -1
```

ตอนที่ 2 ตัวแปรประเภท Strings

Strings เป็นประเภทข้อมูลที่สำคัญและใช้งานทั่วไปในการเขียนโปรแกรม โดย String เป็นลำดับของตัวอักษรหลายตัวเรียงต่อกัน ซึ่งในภาษา Python นั้น String จะอยู่ในเครื่องหมาย Double quote หรือ Single quote เท่านั้น ในตอนนี้ให้คุณคัดลอกโปรแกรมข้างล่างแล้วนำไปรันใน IDE

```
sentent1 = "What's your name?"
sentent2 = 'I\'m a teacher.'
sentent3 = "He said \"I would learn Python first\"."
sentent4 = 'His teach replied "Oh well!"'
print (sentent1)
print (sentent2)
print (sentent3)
print (sentent4)

site = 'ABC' + '.com'
tutorial = 'Python' ' Language'

print(site)
print(tutorial)
```

ผลลัพธ์

```
What's your name?
I'm a teacher.
He said "I would learn Python first".
His teach replied "Oh well!"
ABC.com
Python Language
```

ตอนที่ 3 ตัวแปรประเภท Lists

Lists เป็นประเภทข้อมูลที่มีรูปแบบการจัดเก็บแบบเป็นชุดและลำดับ กล่าวคือมันสามารถเก็บข้อมูลได้หลายค่าในตัวแปรเดียว และมี Index สำหรับเข้าถึงข้อมูล โดยในภาษา Python นั้น List จะเป็นเหมือนอาเรย์ในภาษา C แต่มีความพิเศษกว่าตรงที่สามารถเก็บข้อมูลในประเภทที่แตกต่างกันได้ในตัวแปรเดียวต่อไปนี้ให้คุณคัดลอกโปรแกรมข้างล่างแล้วนำไปรันใน IDE

```
mix = [-1, 2, 3.21, 'k', "Python"]
print(mix)
print("Index at 0 = ", mix[0])
print("Index at 1 = ", mix[1])
print("Index at 2 = ", mix[2])
print("Index at 3 = ", mix[3])
print("Index at 4 = ", mix[4])
mix[3] = "Java"
print(mix)
```

ผลลัพธ์

```
[-1, 2, 3.21, 'k', 'Python']
Index at 0 =  -1
Index at 1 =  2
Index at 2 =  3.21
Index at 3 =  k
Index at 4 =  Python
[-1, 2, 3.21, 'Java', 'Python']
```

ในบทนี้ คุณได้เรียนรู้เกี่ยวกับตัวแปรและประเภทข้อมูลในภาษา Python เราได้พูดถึงการประกาศและการใช้งานตัวแปร รวมถึงข้อมูลประเภทต่างๆ ในภาษา Python เช่น ตัวเลข String และ List เป็นที่เรียบร้อยแล้ว ในบทต่อไปจะเป็นการกล่าวถึงการนำเข้า(Input) และ แสดงผล(Output) ในภาษา Python

บทที่ 4 การรับค่าและแสดงผล

ในบทนี้ คุณจะได้เรียนรู้เกี่ยวกับการรับค่าและการแสดงผล ซึ่งเป็นสิ่งที่ใช้เชื่อมโยงระหว่างโปรแกรมกับผู้ใช้งานโปรแกรม โดยการรับค่าคือการรับข้อมูลจากภายนอกโดยทั่วไปแล้วมักจะเป็นการรับค่าทางคีย์บอร์ด ส่วนการแสดงผลนั้นจะเป็นแสดงข้อความบน Console

ตอนที่ 1 การแสดงข้อความบน Console

การแสดงผลในภาษา Python นั้นจะใช้ฟังก์ชัน print() เพื่อแสดงข้อความ ตัวเลข หรือข้อมูลประเภทอื่น ๆ ออกทางหน้าจอในส่วน Console (Python Shell) ซึ่งคุณสามารถใช้ keyword อาร์กิวเมนต์สำหรับกำหนดการแสดงผลเพื่อกำหนดรูปแบบการแสดงผลของแต่ละอาร์กิวเมนต์ ในตอนนี้ให้คุณคัดลอกโปรแกรมข้างล่างแล้วนำไปรันใน IDE

```
print("Hello Python")
print("A", "B", "C")
print("A" + "B" + "C")
print("One", end=' ')
print("Two", end=' ')
print("Three", end=' ')
```

ผลลัพธ์

```
Hello Python
A B C
ABC
One Two Three
```

ตอนที่ 2 การรับค่าจากคีย์บอร์ด

นอกจากการแสดงผลแล้วนั้น การติดต่อกับผู้ใช้ในอีกรูปแบบหนึ่งคือการรับค่า โดยทั่วไปแล้วมักจะเป็นการรับค่าทางคีย์บอร์ด ในภาษา Python เราใช้ฟังก์ชัน input() สำหรับการรับค่า String จากทางคีย์บอร์ด อย่างไรก็ตาม ในการที่จะรับข้อมูลประเภทอื่น ๆ เช่น Integer หรือ Float เราสามารถใช้ฟังก์ชันที่มากับภาษา Python ในการแปลงข้อมูลจาก String ไปเป็นข้อมูลประเภทอื่นได้ ในตอนนี้ให้คุณคัดลอกโปรแกรมข้างล่างแล้วนำไปรันใน IDE

```
name = input("Enter your name: ")

print("Hello " + name)

a = int(input("Enter first number you need: "))
b = int(input("Enter second number you need: "))

print("a + b = %d" % (a + b))
print("a x b = ", str(int(a) * float(b)))
```

ผลลัพธ์

```
Enter your name: chacharin
Hello chacharin
Enter first number you need: 9
Enter second number you need: 10
a + b = 19
a x b =  90.0
```

ในบทนี้ คุณได้เรียนรู้เกี่ยวกับการรับค่าและการแสดงผล ในภาษา Python เราได้พูดถึงการรับข้อมูลจากภายนอกทางคีย์บอร์ด และแสดงข้อความบน Console ในบทต่อไปจะเป็นการกล่าวถึงตัวดำเนินการ (Operators) ในภาษา Python

บทที่ 5 ตัวดำเนินการ

ในบทนี้คุณจะได้เรียนรู้เกี่ยวกับตัวดำเนินการ (Operators) ซึ่งเป็นกลุ่มของเครื่องหมายหรือสัญลักษณ์ที่ใช้ทำงานเหมือนกับฟังก์ชัน แต่แตกต่างกันตรงไวยากรณ์ โดยในพื้นฐานภาษา Python จะพบการใช้ตัวดำเนินการ 3 ประเภทหลัก ๆ ได้แก่ ตัวดำเนินการทางคณิตศาสตร์ (Arithmetic operators) ตัวดำเนินการเปรียบเทียบ (Comparison operators) และ ตัวดำเนินการตรรกศาสตร์ (Logical operators)

ตอนที่ 1 ตัวดำเนินการทางคณิตศาสตร์

ตัวดำเนินการทางคณิตศาสตร์ (Arithmetic operators) คือตัวดำเนินการที่ใช้สำหรับการคำนวณตัวเลข เช่น การบวก การลบ การคูณ และการหาร มากไปกว่านั้น ในภาษา Python ยังมีตัวดำเนินการทางคณิตศาสตร์เพิ่มเติม เช่น การหารเอาเศษ (Modulo) การหารแบบเลขจำนวนเต็ม และการยกกำลัง เป็นต้น

Operator	Name	Example
+	Addition	a + b
-	Subtraction	a - b
*	Multiplication	a * b
/	Division	a / b
%	Modulo	a % b
**	Power	a ** b

ในตารางข้างต้นนี้ แสดงตัวดำเนินการทางคณิตศาสตร์ประเภทต่าง ๆ สำหรับการคำนวณตัวเลข ด้วยวิธีการต่าง ๆ ในตอนนี้ให้คุณคัดลอกโปรแกรมข้างล่างแล้วนำไปรันใน IDE

```
a, b = 5, 3
print("a + b = ", a + b)
print("a - b = ", a - b)
print("a * b = ", a * b)
print("a / b = ", a / b)
print("a % b = ", a % b)
print("a ** b = ", a ** b)
```

ผลลัพธ์

```
a + b =  8
a - b =  2
a * b =  15
a / b =  1.6666666666666667
a % b =  2
a ** b =  125
```

ตอนที่ 2 ตัวดำเนินการเปรียบเทียบ

ตัวดำเนินการเปรียบเทียบ (Comparison operators) คือตัวดำเนินการที่ใช้สำหรับเปรียบเทียบตัวเลข หรือ ค่าในตัวแปร ซึ่งผลลัพธ์ของการเปรียบเทียบนั้นจะเป็น True (1) หากเงื่อนไขเป็นจริง และเป็น False (0) หากเงื่อนไขไม่เป็นจริง ตัวดำเนินการเปรียบเทียบมักจะใช้กับคำสั่งตรวจสอบเงื่อนไข if และคำสั่งวนซ้ำ for หรือ while เพื่อควบคุมการทำงานของโปรแกรม

Operator	Name	Example
<	Less than	a < b
<=	Less than or equal	a <= b
>	Greater than	a > b
>=	Greater than or equal	a >= b
==	Equal	a == b
!=	Not equal	a != b

ในตารางข้างต้นนี้ แสดงตัวดำเนินการเปรียบเทียบประเภทต่าง ๆ เช่น การเปรียบเทียบความเท่ากัน หรือการเปรียบเทียบค่ามากกว่าหรือน้อยกว่า ในตอนนี้ให้คุณคัดลอกโปรแกรมข้างล่างแล้วนำไปรันใน IDE

```
print('4 = 4? :', 4 == 4)
print('3 > 10? :', 3 > 10)

a = 10
b = 8

print('a != b? :', a != b)
print('a - b = 2? :', a - b == 2)
```

ผลลัพธ์

```
4 = 4? : True
3 > 10? : False
a != b? : True
a - b = 2? : True
```

ตอนที่ 3 ตัวดำเนินการตรรกศาสตร์

ตัวดำเนินการตรรกศาสตร์ (Logical operators) คือตัวดำเนินการที่ใช้สำหรับประเมินค่าทางตรรกศาสตร์ ซึ่งมีเพียงจริง (True) และเท็จ (False) เท่านั้น

Operator	Example	Result
and	a and b	True if a and b are true, else False
or	a or b	True if a or b are true, else False
not	not a	True if a is False, else True

ในภาษา Python นั้นมีตัวดำเนินการทางตรรกศาสตร์ 3 ชนิด คือ ตัวดำเนินการ and ตัวดำเนินการ or และตัวดำเนินการ not ในตอนนี้ให้คุณคัดลอกโปรแกรมข้างล่างแล้วนำไปรันใน IDE

```
username = input('Username: ')
password = input('Password: ')
if (username == 'admin' and password == '1234'):
        print(' User and Password is True ')
elif (username == 'admin' or password == '1234'):
        print(' User or Password is False ')
else:
        print('Invalid username or password.')
```

ผลลัพธ์

```
Username: admin
Password: 1234
 User and Password is True


Username: admin
Password: 9999
 User or Password is False


Username: admin
Password: 9999
 User or Password is False
```

ในบทนี้ คุณได้เรียนรู้เกี่ยวกับ ตัวดำเนินการ 3 ประเภทหลัก ตัวดำเนินการทางคณิตศาสตร์ (Arithmetic operators) ตัวดำเนินการเปรียบเทียบ (Comparison operators) และ ตัวดำเนินการตรรกศาสตร์ (Logical operators) เป็นที่เรียบร้อยแล้ว ต่อไปคุณจะได้เรียนรู้กับการตั้งเงื่อนไขของโปรแกรม

บทที่ 6 คำสั่งเงื่อนไข

ในบทนี้คุณจะได้เรียนรู้เกี่ยวกับคำสั่งเงื่อนไขในภาษา Python เราจะพูดถึงการควบคุมการทำงานโปรแกรมด้วยคำสั่ง if, if else และ elif เพื่อให้โปรแกรมสามารถทำงานซับซ้อนและมีประสิทธิภาพมากขึ้น

ตอนที่ 1 คำสั่ง if

คำสั่ง if เป็นคำสั่งที่ใช้ควบคุมการทำงานของโปรแกรมที่เป็นพื้นฐานและง่ายที่สุด เราใช้คำสั่ง if เพื่อสร้างเงื่อนไขให้โปรแกรมทำงานตามที่เราต้องการเมื่อเงื่อนไขนั้นตรงกับที่เรากำหนด เช่น การตรวจสอบค่าในตัวแปรกับตัวดำเนินการประเภทต่าง ๆ ในตอนนี้ให้คุณคัดลอกโปรแกรมข้างล่างแล้วนำไปรันใน IDE

```
n = 10
if n == 10:
        print('n equal to 10')

logged_in = False

if not logged_in:
        print('You must login to continue')

m = 4
if m % 2 == 0 and m > 0:
        print('m is even and positive numbers')

if 3 > 10:
        print('This block isn\'t executed')
```

ผลลัพธ์

```
n equal to 10
You must login to continue
m is even and positive numbers
```

ตอนที่ 2 คำสั่ง if else

หลังจากที่คุณได้รู้จักกับคำสั่ง if ไปแล้ว อีกคำสั่งหนึ่งที่ทำงานควบคู่กับคำสั่ง if คือคำสั่ง else clause โดยโปรแกรมจะทำงานในคำสั่ง else ถ้าหากเงื่อนไขในคำสั่ง if นั้นไม่เป็นจริง กล่าวอีกนัยหนึ่งคือ มันจะทำงานเมื่อเงื่อนไขก่อนหน้านั้นไม่เป็นจริง ในตอนนี้ให้คุณคัดลอกโปรแกรมข้างล่างแล้วนำไปรันใน IDE

```
n = 5
if n == 10:
        print('n equal to 10')
else:
        print('n is something else except 10')

name = 'B'
if name == 'B':
        print('Hi, B.')
else:
        print('Who are you?')

money = 300
if money >= 350:
        print('You can buy an iPad')
else:
        print('You don\'t have enough money to buy an iPad')
```

ผลลัพธ์

```
n is something else except 10
Hi, B.
You don't have enough money to buy an iPad
```

ตอนที่ 3 คำสั่ง if elif

คำสั่ง elif นั้นเป็นคำสั่งที่ใช้สำหรับสร้างเงื่อนไขแบบหลายทางเลือกให้กับโปรแกรมที่มีการทำงานเช่นเดียวกับ switch case ในภาษาอื่น ๆ คำสั่ง elif นั้นจำเป็นต้องใช้หลังจากคำสั่ง if เสมอ และสามารถมี else ได้ในเงื่อนไขสุดท้าย ในตอนนี้ให้คุณคัดลอกโปรแกรมข้างล่างแล้วนำไปรันใน IDE

```
print('Welcome to Robot\'s game')
level = input('Enter level (1 - 4): ')

if level == '1':
        print('Easy')
elif level == '2':
        print('Medium')
elif level == '3':
        print('Hard')
elif level == '4':
        print('Expert')
else:
        print('Invalid level selected')
```

ผลลัพธ์

```
Welcome to Robot's game
Enter level (1 - 4): 1
Easy

Welcome to Robot's game
Enter level (1 - 4): 4
Expert

Welcome to Robot's game
Enter level (1 - 4): 9
Invalid level selected
```

ในบทนี้คุณได้เรียนรู้เกี่ยวกับคำสั่งเงื่อนไขซึ่งประกอบไปด้วยคำสั่ง if, if else และ elif เป็นที่เรียบร้อยแล้ว ต่อไปคุณจะได้เรียนรู้เกี่ยวกับการสร้างโปรแกรมแบบวนซ้ำ

บทที่ 7 คำสั่งวนซ้ำ

ในบทนี้คุณจะได้เรียนรู้เกี่ยวกับการควบคุมการทำงานของโปรแกรมโดยการใช้คำสั่ง while loop และ for loop คำสั่งเหล่านี้สามารถควบคุมโปรแกรมให้ทำงานซ้ำ ๆ ในเงื่อนไขที่กำหนด

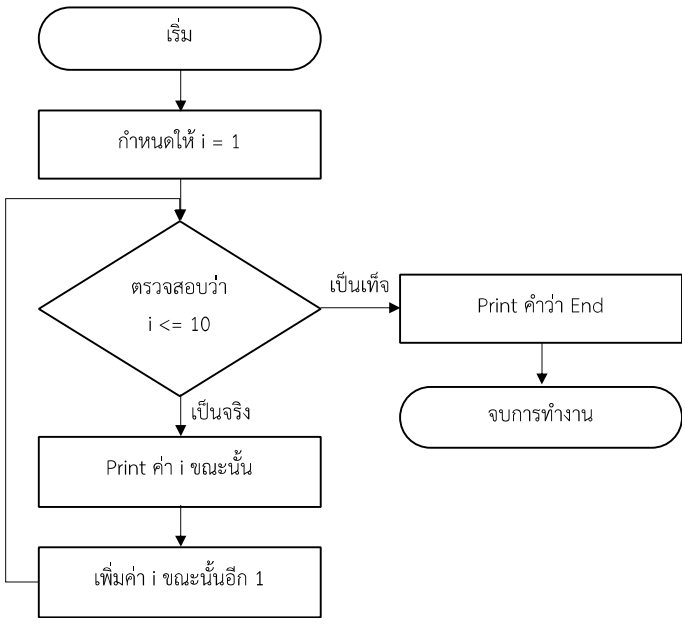
ตอนที่ 1 คำสั่ง while loop

คำสั่ง while loop เป็นคำสั่งวนซ้ำที่ง่ายและพื้นฐานที่สุดในภาษา Python คำสั่ง while loop นั้นใช้ควบคุมโปรแกรมให้ทำงานบางอย่างซ้ำ ๆ ถ้าหากเงื่อนไขของ while loop นั้นยังคงเป็นจริงอยู่ ในตอนนี้ให้คุณคัดลอกโปรแกรมข้างล่างแล้วนำไปรันใน IDE

```
i = 1
while i <= 10:
        print(i, end = ', ')
        i = i + 1
print('END')
```

ผลลัพธ์

```
1, 2, 3, 4, 5, 6, 7, 8, 9, 10, END
```



รูปแสดงผังงาน (Flowchart) การทำงานของ while loop

ตอนที่ 2 คำสั่ง for loop

คำสั่ง for loop เป็นคำสั่งวนซ้ำที่ใช้ควบคุมการทำงานซ้ำ ๆ ในจำนวนรอบที่แน่นอน มักจะใช้สำหรับการวนอ่านค่าภายในฟังก์ชัน range() ในตอนนี้ให้คุณคัดลอกโปรแกรมข้างล่างแล้วนำไปรันใน IDE

```
for i in range(5):
        print(i, end = ', ')
print('END')

for i in range(3, 6):
        print(i, end = ', ')
print('END')

for i in range(3, 8, 2):
        print(i, end = ', ')
print('END')
```

ผลลัพธ์

```
0, 1, 2, 3, 4, END
3, 4, 5, END
3, 5, 7, END
```

ในบทนี้คุณจะได้เรียนรู้เกี่ยวกับการควบคุมการทำงานของโปรแกรมโดยการใช้คำสั่ง while loop และ for loop เป็นที่เรียบร้อยแล้ว ต่อไปจะเป็นเรื่องของ ฟังก์ชัน ซึ่งเป็นบทสุดท้ายที่เป็นพื้นฐานของภาษา Python

บทที่ 8 ฟังก์ชัน

ในบทนี้คุณจะได้เรียนรู้เกี่ยวกับฟังก์ชันในภาษา Python โดยเราจะพูดถึงการสร้างและการใช้งานฟังก์ชันในเบื้องต้น เช่น Default Argument และ Keyword Augment

ตอนที่ 1 การสร้างและเรียกใช้ฟังก์ชัน

ฟังก์ชัน (Function) คือส่วนของโปรแกรมที่ทำงานเพื่อวัตถุประสงค์บางอย่าง โดยในภาษา Python คุณสามารถสร้างฟังก์ชันของคุณได้เองเพื่อให้โปรแกรมทำงานตามที่ต้องการ ในการเขียนโปรแกรมเรามักจะแยกโค้ดที่มีการทำงานเหมือนๆ กันเป็นฟังก์ชันเอาไว้ และเรียกใช้ฟังก์ชันนั้นซ้ำ ๆ ซึ่งเป็นแนวคิดของการนำโค้ดกลับมาใช้ใหม่ (Code reuse) ในตอนนี้ให้คุณคัดลอกโปรแกรมข้างล่างแล้วนำไปรันใน IDE

```
def hello(name):
        print('Hello, '+name)

def area(width, height):
        c = width * height
        return c

hello('Robotics')

print('Area 5x5 = ',area(5, 5))
```

ผลลัพธ์

```
Hello, Robotics
Area 5x5 =  25
```

ตอนที่ 2 Default Argument Values

ในภาษา Python เราสามารถสร้างฟังก์ชันโดยการกำหนด Default Argument ให้กับฟังก์ชันได้ โดย Default Argument เป็นการการกำหนดค่าเริ่มต้นให้กับอาร์กิวเมนต์ที่ส่งเข้ามายังฟังก์ชัน นั่นทำให้เราสามารถเรียกใช้งานฟังก์ชันโดยการส่งอาร์กิวเมนต์น้อยกว่าจำนวนที่กำหนดไว้ในฟังก์ชันได้ ซึ่งอำนวยความสะดวกในการใช้งานมากขึ้น มาดูตัวอย่างการสร้างและใช้งานฟังก์ชันกับ Default Argument ด้วยการคัดลอกโปรแกรมข้างล่างแล้วนำไปรันใน IDE

```
def show_info(name, salary = 999, lang = "Python"):
        print('Name: %s' % name)
        print('Salary: %d' % salary)
        print('Language: %s' % lang)
        print()

show_info('Robobotic')
show_info('Robot', 200)
show_info('Programing', 300, 'Java')
```

ผลลัพธ์

```
Name: Robobotic
Salary: 999
Language: Python

Name: Robot
Salary: 200
Language: Python

Name: Programing
Salary: 300
Language: Java
```

ตอนที่ 3 Keyword Arguments

ในภาษา Python เราสามารถเรียกใช้งานฟังก์ชันในรูปแบบของ Keyword Argument โดยการใช้ชื่อของพารามิเตอร์สำหรับส่งอาร์กิวเมนต์ได้โดยตรง ซึ่งในการใช้งานนั้นพารามิเตอร์ต้องมีการกำหนดในรูปแบบของ Default Argument ก่อนเสมอ มาดูตัวอย่างการใช้งาน Keyword Arguments ด้วยการคัดลอกโปรแกรมข้างล่างแล้วนำไปรันใน IDE

```
def create_button(id, color = '#ffffff', text = 'Button', size =
16):
        print('Button ID: %d' % id)
        print('Attributes:')
        print('Color: %s' % color)
        print('Text: %s' % text)
        print('Size: %d px' % size)
        print()

create_button(10)
create_button(11, color = '#4286f4', text = 'Sign up')
create_button(id = 12, color = '#323f54', size = 24)
```

ผลลัพธ์

```
Button ID: 10
Attributes:
Color: #ffffff
Text: Button
Size: 16 px

Button ID: 11
Attributes:
Color: #4286f4
Text: Sign up
Size: 16 px

Button ID: 12
Attributes:
Color: #323f54
Text: Button
Size: 24 px
```

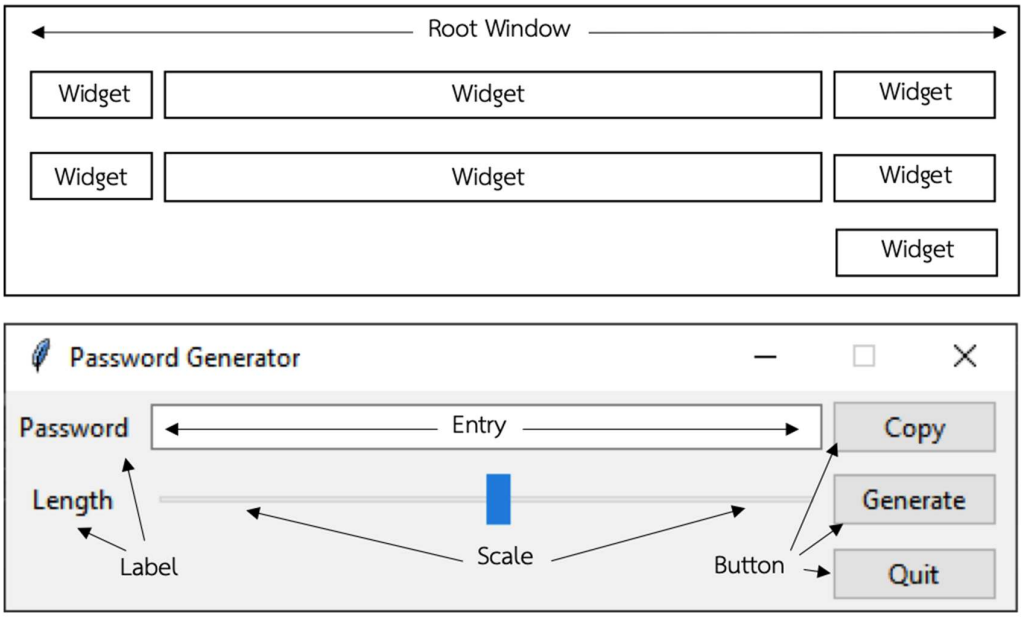
ในบทนี้คุณจะได้เรียนรู้การสร้างและการใช้งานฟังก์ชันในภาษา Python เป็นที่เรียบร้อยแล้ว ถือเป็นอันจบเนื้อหาพื้นฐานการเขียนโปรแกรมภาษา Python โดยสมบูรณ์ ต่อไปจะเป็นการประยุกต์ใช้ภาษา Python ในการทำงานเกี่ยวกับการควบคุมหุ่นยนต์

บทที่ 9 ส่วนต่อประสานกับผู้ใช้ด้วยกราฟฟิก

ในบทนี้คุณจะได้เรียนรู้เกี่ยวกับส่วนต่อประสานกับผู้ใช้ด้วยกราฟฟิก (Graphical User Interface) หรือเรียกตัวย่อว่า GUI โดยในภาษา Python มีโมดูล Tkinter ที่ย่อมาจาก Toolkit Interface ซึ่งเป็นชุดคำสั่งสำหรับการสร้าง GUI ที่ติดมากับ Python อยู่แล้ว ซึ่งในบทเรียนนี้คุณจะได้เรียนรู้วิธีการใช้โมดูลนี้

ตอนที่ 1 โครงสร้างของ GUI

ก่อนที่จะสร้าง GUI คุณจำเป็นจะต้องเข้าใจองค์ประกอบของ GUI อย่างง่ายที่สุดตามภาพข้างล่างนี้



องค์ประกอบหลักของการสร้าง GUI ประกอบไปด้วย 2 องค์ประกอบหลัก ๆ ได้แก่

1) Root Window คือ กรอบของส่วนที่ต้องการให้แสดงผล

2) Widget คือ ส่วนประกอบย่อย ๆ ในการแสดงผล ซึ่งในบทเรียนนี้จะพูดถึง Widget 4 ชนิด ได้แก่

2.1) Label คือ พื้นที่ว่างสำหรับแสดงผลข้อความ สัญลักษณ์ หรือ รูปภาพ

2.2) Button คือ ปุ่มกด

2.3) Entry คือ ช่องสำหรับกรอกข้อความ หรือ ตัวเลข

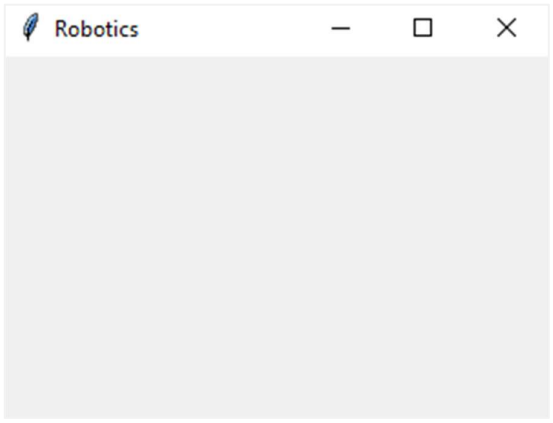
2.4) Scale คือ แถบเลื่อน

ตอนที่ 2 Root window

Root window คือ กรอบของส่วนที่ต้องการให้แสดงผล(หน้าต่าง) ซึ่งเป็นหัวใจหลักของ GUI โดยก่อนจะเริ่มการสร้างคุณต้องเรียกใช้ โมดูลคำสั่ง tkinter บน Python เสียก่อน จากนั้นคุณจะสามารถกำหนดขนาดของหน้าต่างได้ด้วยคำสั่ง geometry และสามารถกำหนดชื่อของหน้าต่างได้ด้วยคำสั่ง title ในตอนนี้ให้คุณคัดลอกโปรแกรมข้างล่างแล้วนำไปรันใน IDE

```
from tkinter import*
window = Tk()
window.geometry("300x200")
window.title("Robotics")
```

ผลลัพธ์

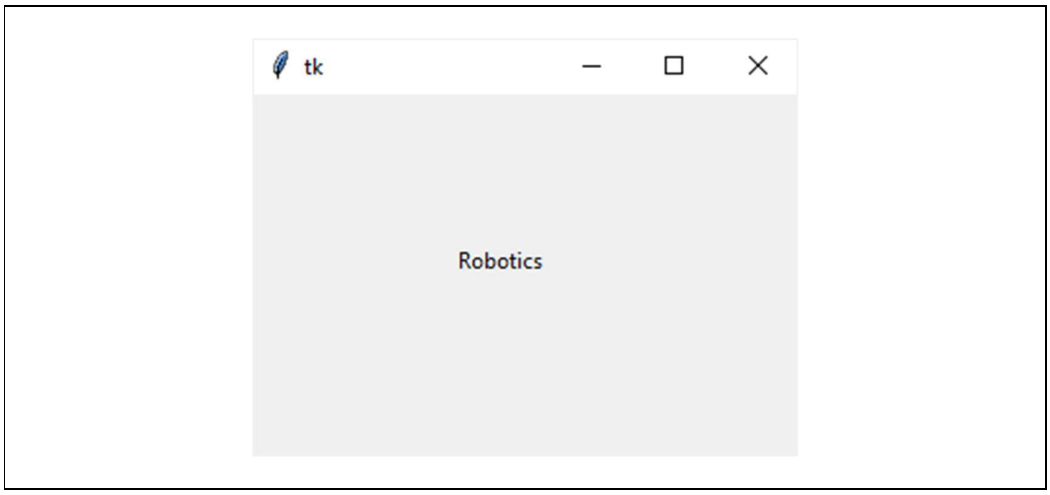


ตอนที่ 3 Widget Label

Label คือ พื้นที่ว่างสำหรับแสดงผลข้อความ สัญลักษณ์ หรือ รูปภาพ ทั้งนี้ ในบทเรียนนี้จะพูดถึงการแสดงชุดข้อความบนหน้าต่าง โดยวิธีการสร้าง Widget สามารถทำได้โดยใช้คำสั่งเรียกประเภทของ Widget และกำหนดคุณสมบัติของ Widget นั้น และ หลังจากนั้นให้กำหนดว่าจะวาง Widget นั้นลงบนพิกัดใดของหน้าต่าง ในตอนนี้ให้คุณคัดลอกโปรแกรมข้างล่างแล้วนำไปรันใน IDE

```
from tkinter import*
window = Tk()
window.geometry("300x200")
L1 = Label(window, text = "Robotics")
L1.place(x=110,y=80)
```

ผลลัพธ์

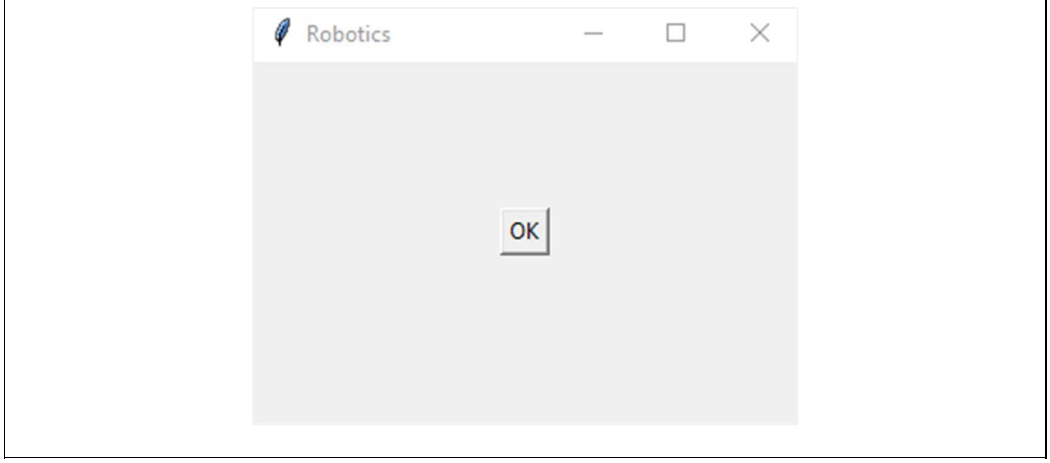


ตอนที่ 4 Widget Button

Button คือ ปุ่มกด และทุก ๆ การกด จะเกิดงาน(Command Event) ขึ้นในระบบ ซึ่งคุณสามารถสร้างชุดของฟังก์ชันเพื่อให้ระบบดำเนินการเมื่อถูกกดปุ่มได้ ในตอนนี้ให้คุณคัดลอกโปรแกรมข้างล่างแล้วนำไปรันใน IDE

```
from tkinter import*
window = Tk()
window.geometry("300x200")
window.title("Robotics")

def ok():
    print("OK")

B1 = Button(window, text= "OK", command=ok)
B1.place(x=135,y=80)
```

ผลลัพธ์

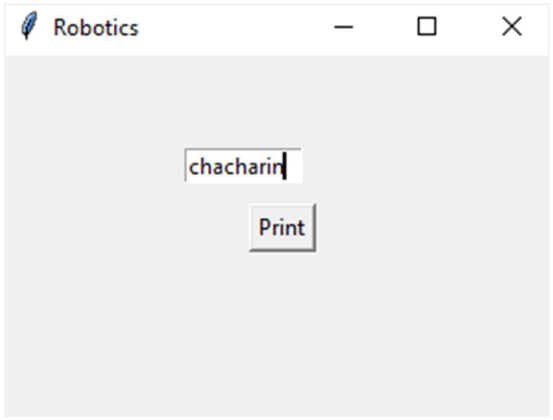


ตอนที่ 5 Widget Entry ร่วมกับ Button

Entry คือ ช่องสำหรับกรอกข้อความ โดยมากแล้วจะใช้ร่วมกับปุ่มกดเพื่อสั่งให้ระบบนำเข้าข้อความ ผ่านคำสั่ง get() ในตอนนี้ให้คุณคัดลอกโปรแกรมข้างล่างแล้วนำไปรันใน IDE

```
from tkinter import*
window = Tk()
window.geometry("300x200")
window.title("Robotics")

def Print():
    text_value = E1.get()
    print(text_value)

E1 = Entry(window, width=10)
E1.place(x=100,y=50)

B1 = Button(window, text= "Print", command=Print)
B1.place(x=135,y=80)
```

ผลลัพธ์



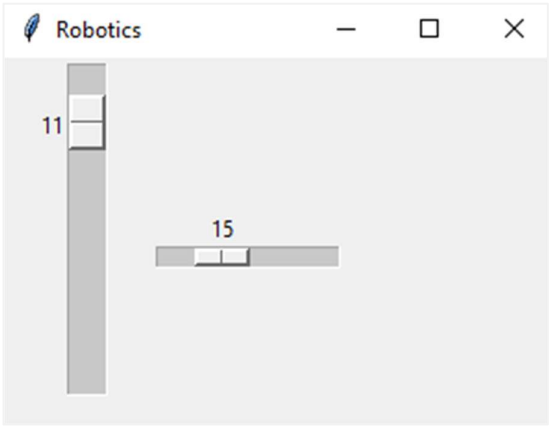

```
chacharin
```

ตอนที่ 6 Widget Scale

Scale คือ แถบเลื่อน และทุก ๆ การเลื่อนจะเกิดงาน (Command Event) ขึ้นในระบบ โดยค่าตัวเลขใน Scale จะถูกเก็บไว้ในตัวแปรที่เป็นตัวแปรตั้งต้น (val) ดังนั้นคุณสามารถนำค่าตัวเลขของ Scale มาใช้ได้ด้วยคำสั่ง get() เพื่ออ่านค่าจาก val ของ Scale นั้น ๆ ในตอนนี้ให้คุณคัดลอกโปรแกรมข้างล่างแล้วนำไปรันใน IDE

```
from tkinter import*
window = Tk()
window.geometry("300x200")
window.title("Robotics")

def print_s1(val):
    scale_value1 = S1.get()
    print("Scale1: "+ str(scale_value1))

def print_s2(val):
    scale_value2 = S2.get()
    print("Scale2: "+ str(scale_value2))

S1 = Scale(window, from_=0, to=100,width=20, length=180, orient =
VERTICAL, command=print_s1)
S1.place(x=10, y=0)

S2 = Scale(window, from_=0, to=50,width=10, length=100,orient =
HORIZONTAL, command=print_s2)
S2.place(x=80, y=80)
```

ผลลัพธ์



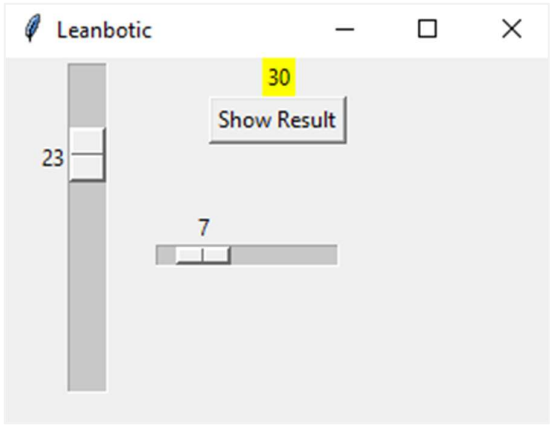

```
Scale1: 1
Scale1: 2
Scale1: 4
Scale1: 5
Scale1: 6
Scale1: 7
Scale1: 9
Scale1: 10
Scale1: 11
Scale2: 3
Scale2: 4
Scale2: 7
Scale2: 9
Scale2: 11
Scale2: 12
Scale2: 13
Scale2: 14
Scale2: 15
```

ตอนที่ 7 การทำงานร่วมกันหลาย Widget

หลังจากที่คุณได้เรียนรู้เกี่ยวกับการใช้ Widget หลากหลายประเภทไปแล้ว ในตอนที่ 7 นี้ จะเป็นการประยุกต์ใช้ความรู้หลายเรื่องเข้าด้วยกัน

```
from tkinter import*
window = Tk()
window.geometry("300x200")
window.title("Leanbotic")

def print_s1(val):
    global scale_value1
    scale_value1 = S1.get()

def print_s2(val):
    global scale_value2
    scale_value2 = S2.get()

def change_label():
    global scale_value1
    global scale_value2
    L1.configure(text=scale_value1 + scale_value2)
    L1.update()

L1 = Label(window, text=0, bg="yellow")
L1.pack()
B1 = Button(window,text="Show Result",command=change_label)
B1.pack()
S1 = Scale(window, from_=0, to=100,width=20, length=180, orient =
VERTICAL, command=print_s1)
S1.place(x=10, y=0)
S2 = Scale(window, from_=0, to=50,width=10, length=100,orient =
HORIZONTAL, command=print_s2)
S2.place(x=80, y=80)
```

ผลลัพธ์



บทที่ 10 การเชื่อมต่อ Microcontroller

ในบทนี้คุณจะได้เรียนรู้เกี่ยวกับโมดูล pyFirmata ซึ่งเป็นโมดูลสื่อสารผ่าน Protocol(ขั้นตอน) มาตรฐานที่ถูกสร้างขึ้นไว้สำหรับสั่งการ Pin Analog และ Pin Digital ของบอร์ด Microcontroller Arduino

ตอนที่ 1 การติดตั้งโมดูล pyFirmata

ก่อนที่จะเริ่มใช้ชุดคำสั่งต่าง ๆ ในโมดูล pyFirmata คุณจำเป็นจะต้องติดตั้ง โมดูลดังกล่าวลงในเครื่องคอมพิวเตอร์ของคุณ โดยมีลำดับขั้นตอนดังนี้

1) เปิด Command Prompt (CMD) ของ Microsoft Windows

2) เรียกใช้คำสั่ง pip (C:\Users\.....>pip) ใน CMD แล้วคุณจะพบรายการคำสั่งของ pip ทั้งหมด

3) เรียใช้คำสั่ง pip install pyFirmata (C:\Users\.....>pip install pyFirmata)

4) รอจนกระทั้ง ระบบจะแจ้งว่า Successfully installed pyFirmata... เป็นอันสำเร็จ

ตอนที่ 2 ติดตั้ง Standard Firmata บนบอร์ด Arduino

ก่อนที่จะเริ่มควบคุมบอร์ด Arduino ด้วย pyFirmata คุณจำเป็นต้องบรรจุชุดคำสั่ง Standard Firmata ในภาษา C++ บน Arduino ผ่าน Arduino IDE ก่อน โดยมีลำดับขั้นตอนดังนี้

1) ดาวน์โหลดและติดตั้ง Arduino IDE ลงบนคอมพิวเตอร์ของคุณ

2) เชื่อมสาย USB ระหว่างบอร์ด Arduino กับ คอมพิวเตอร์

3) เปิด Software Arduino IDE จากนั้น เลือกชนิดบอร์ดและกำหนดหมายเลข COM port

4) เลือกเมนู File > Example > Firmata > Standard Firmata

5) อัพโหลดชุดคำสั่งลงบอร์ด Arduino เป็นอันสำเร็จ

ตอนที่ 3 ทดสอบ Protocol Firmata

คุณสามารถทำการทดสอบ Protocol หลังจากที่บรรจุชุดคำสั่ง Standard Firmata แล้ว โดยการใช้ Software สำเร็จรูปที่สามารถดาวน์โหลดมาใช้ได้ฟรี มีขั้นตอนการทดสอบ Protocol ดังนี้

1) ดาวน์โหลดลิ้งก์ http://www.pjrc.com/teensy/firmata_test/firmata_test.exe

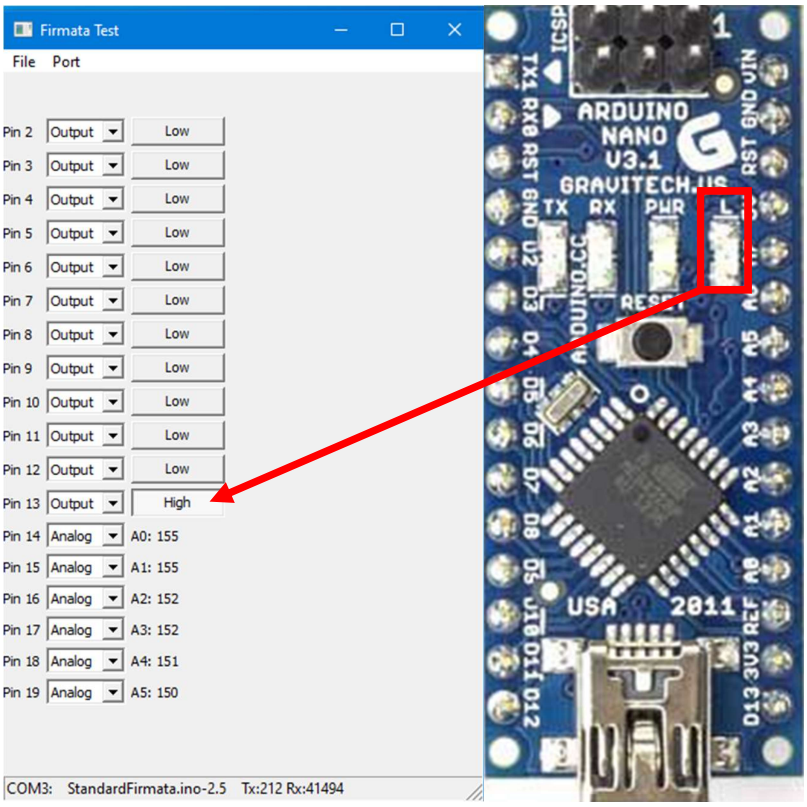
2) เปิดไฟล์ firmata_test.exe ที่ดาวน์โหลดสำเร็จ จากนั้นให้คุณกำหนดหมายเลข COM port

3) ทดลองกดปุ่ม Low สลับกับ High ที่ ตำแหน่ง Pin 13

4) ในระหว่างที่กดปุ่มให้สังเกต LED ดวงเล็ก ๆ บนบอร์ด Arduino

5) หาก LED ดวงดังกล่าว ติด/ดับ ตามการกดปุ่ม High สลับกับ Low เป็นอันสมบูรณ์

6) หาก LED ดวงดังกล่าวไม่มีการติด / ดับตามการกดปุ่ม High สลับกับ Low ให้คุณกลับไปทำตอนติดตั้ง Standard Firmata บนบอร์ด Arduino ใหม่อีกครั้ง



รูปแสดงการใช้ซอฟต์แวร์ Firmata Test กับบอร์ด Arduino Nano

บทที่ 11 การเขียนโปรแกรมควบคุมหุ่นยนต์

ในบทนี้คุณจะได้เรียนรู้เกี่ยวกับรายละเอียดของคำสั่งเพื่อควบคุมมอเตอร์หุ่นยนต์แขนกลผ่านโมดูล pyFirmata ซึ่งคุณได้ทำการติดตั้งและทดสอบการสื่อสารระหว่างคอมพิวเตอร์ และ บอร์ด Arduino ไปแล้ว

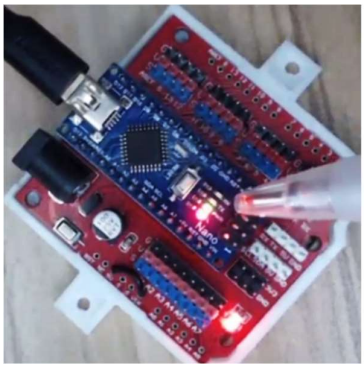
ตอนที่ 1 การควบคุม Output LED

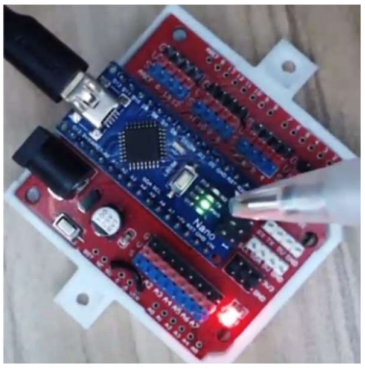
LED ย่อมาจาก Light Emitting Diode ซึ่งหมายถึง ไดโอดชนิดเปล่งแสง มีคุณสมบัติในการส่องแสงออกมาเมื่อมีกระแสไฟฟ้าไหลผ่าน ซึ่งในบอร์ด Arduino ของคุณจะมี LED ติดตั้งอยู่ควบคู่กับ Pin Digital 13 ในตอนนี้ให้คุณคัดลอกโปรแกรมข้างล่างแล้วนำไปรันใน IDE เพื่อทดลองควบคุมการเปิด/ปิด LED

```
from pyfirmata import Arduino
import time
board = Arduino('COM…')
print("Communication Successfully")

led13 = board.get_pin('d:13:o')

while(True):
    led13.write(1)
    time.sleep(1)
    led13.write(0)
    time.sleep(1)
```

ผลลัพธ์

เปิด LED

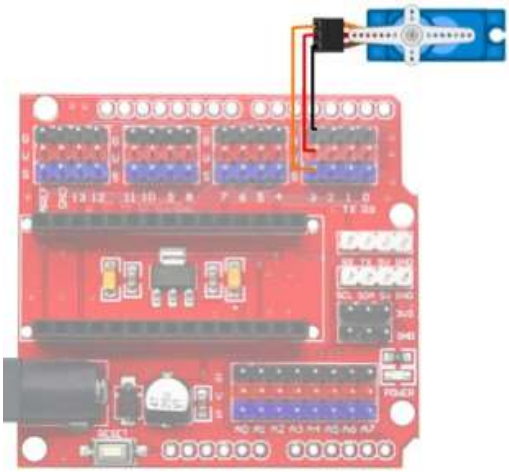




ปิด LED

ตอนที่ 2 การควบคุม Servo 180 degree แบบกำหนดค่าเป้าหมาย

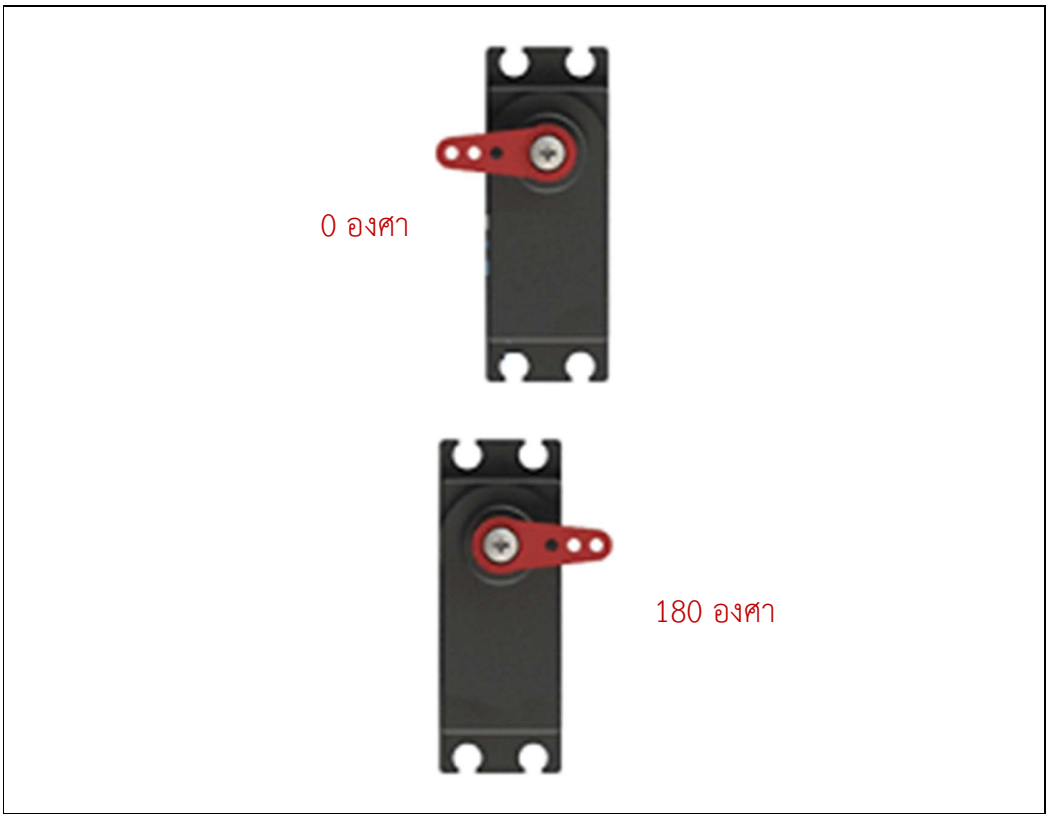
Servo 180-degree เป็นมอเตอร์ชนิดหนึ่งที่คุณสามารถควบคุมองศาของการหมุนได้ในขอบเขต 0 – 180 องศา ด้วยการให้สัญญาณ PWM ไปยัง จุด Signal(สายสีส้ม/สีน้ำตาล) แล้ววงจรภายในมอเตอร์จะกำหนดปริมาณและทิศทางของกระแสไฟฟ้า เพื่อให้มอเตอร์หมุนไปหยุดยังองศาที่กำหนด โดยในชุดหุ่นยนต์นี้มอเตอร์ต้องการไฟเลี้ยงที่มีแรงดันขนาด 5 โวลต์ (สายสีแดง) และทั้งวงจรจะต้องถูกอ้างอิงด้วยจุด Ground (สายสีดำ) ร่วมกัน ทั้งนี้คุณสามารถใช้ โมดูล pyFirmata เพื่อกำหนดเป้าหมายขององศาที่ต้องการให้มอเตอร์นี้ไปหยุด ณ ตำแหน่งใด ๆ ในช่วง 0 – 180 องศา ซึ่งในตอนนี้ให้คุณคัดลอกโปรแกรมข้างล่างแล้วนำไปรันใน IDE พร้อมทั้งต่อสายมอเตอร์เข้าที่ Pin Digital 3 ดังภาพ เพื่อทดลองการกำหนดองศาของมอเตอร์



รูปแสดงการต่อมอเตอร์ Servo 180 degree เข้ากับบอร์ดขยาย Arduino Nano

```
from pyfirmata import Arduino
import time
board = Arduino('COM…')
print("Communication Successfully")

Servo3 = board.get_pin('d:3:s')

while(True):
    Servo3.write(0)
    time.sleep(1)
    Servo3.write(180)
    time.sleep(1)
```

ผลลัพธ์

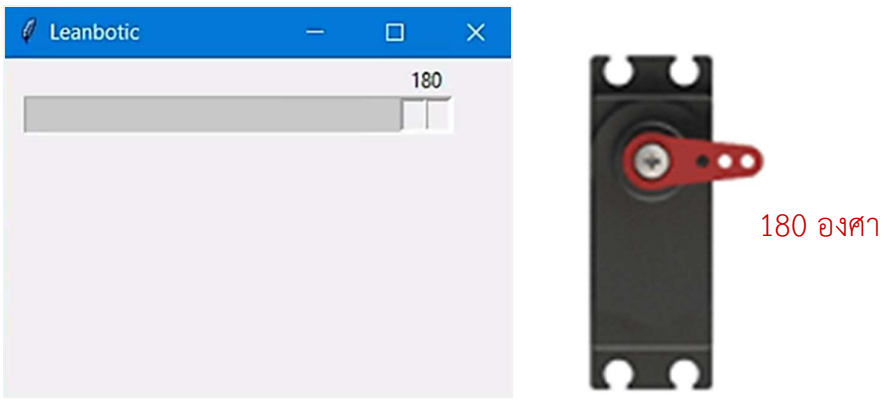


ตอนที่ 3 การใช้ Widget Scale ควบคุม Servo 180 degree

หลังจากที่คุณได้เรียนรู้ชุดคำสั่งเกี่ยวกับการสร้าง GUI รวมถึงการควบคุมองศาของมอเตอร์ไปแล้ว ในตอนนี้จะเป็นการประยุกต์ใช้ความรู้ทั้ง 2 เรื่อง เพื่อการสร้าง Widget Scale ขึ้นมาควบคุม Servo 180 degree ในตอนนี้ให้คุณคัดลอกโปรแกรมข้างล่างแล้วนำไปรันใน IDE

```
from pyfirmata import Arduino
import time
board = Arduino('COM…')
print("Communication Successfully")
Servo3 = board.get_pin('d:3:s')

from tkinter import*
window = Tk()
window.geometry("300x200")
window.title("Leanbotic")

def print_s1(val):

    scale_value1 = S1.get()
    print("Scale1: "+ str(scale_value1))
    Servo3.write(scale_value1)


S1 = Scale(window, from_=0, to=180,width=20, length=250, orient =
HORIZONTAL, command=print_s1)
S1.place(x=10, y=0)
```

ผลลัพธ์

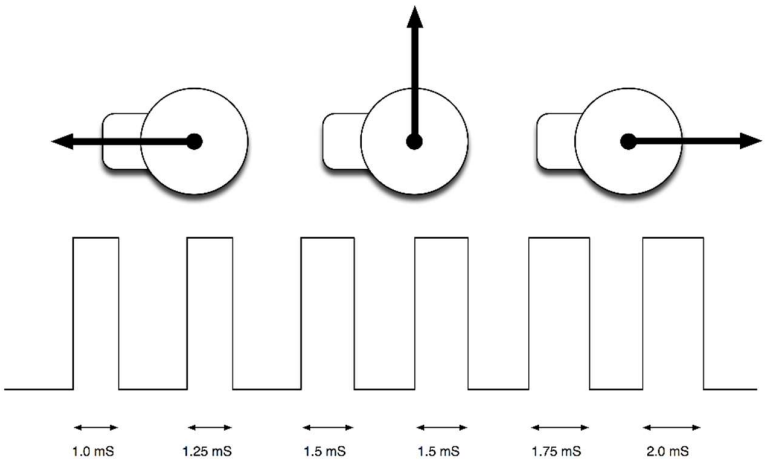


ตอนที่ 4 การควบคุม Servo 180 degree แบบปรับละเอียด

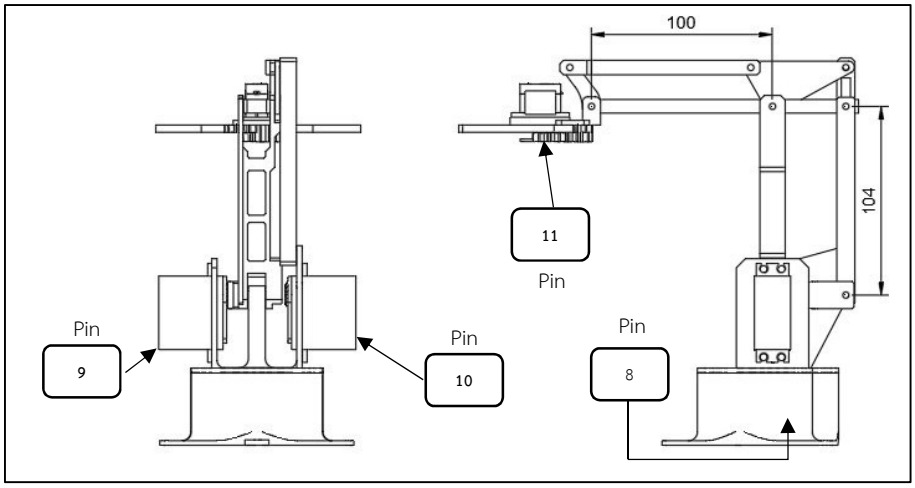
ในตอนที่ 2 คุณได้ทดลองควบคุมองศาของ Servo 180 degree โดยการกำหนดเป้าหมายขององศาที่ต้องการให้มอเตอร์ไปหยุด ณ ตำแหน่งใด ๆ ในช่วง 0 – 180 องศาไปแล้ว ซึ่งคุณจะพบว่ามอเตอร์จะหมุนไปยังจุดต่างในลักษณะการกระชากตัวไปอย่างรวดเร็ว ดังนั้นในตอนที่ 4 นี้จะเป็นเทคนิคการเขียนโปรแกรมเพื่อให้มอเตอร์เคลื่อนที่ไปยังจุดที่เป็นเป้าหมายอย่างช้า ๆ ทีละองศา เพื่อนำไปประยุกต์ใช้ตามความเหมาะสมในการใช้งานตามความต้องการของงานแต่ละประเภทที่แตกต่างกันไป ตอนนี้ให้คุณคัดลอกโปรแกรมข้างล่างแล้วนำไปรันใน IDE

```
from pyfirmata import Arduino
import time
board = Arduino('COM…')
print("Communication Successfully")

Servo3 = board.get_pin('d:3:s')
Servo3.write(0)
time.sleep(1)

position=0
while(position<=180):
    Servo3.write(position)
    position = position+1
    time.sleep(0.008)

position=180
while(position>=0):
    Servo3.write(position)
    position = position-1
    time.sleep(0.008)
```

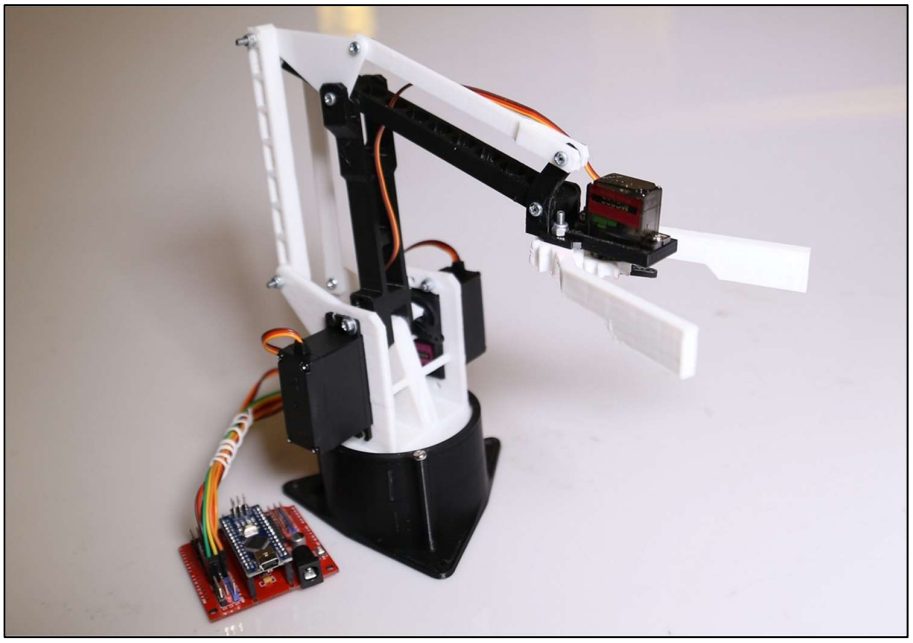
ผลลัพธ์



ตอนที่ 5 การควบคุมท่าทางแขนกล



รูปแสดงหมายเลข Pin ที่มอเตอร์แต่ละตัวของแขนกลที่จะใช้ในตัวอย่างโปรแกรมตอนที่ 5



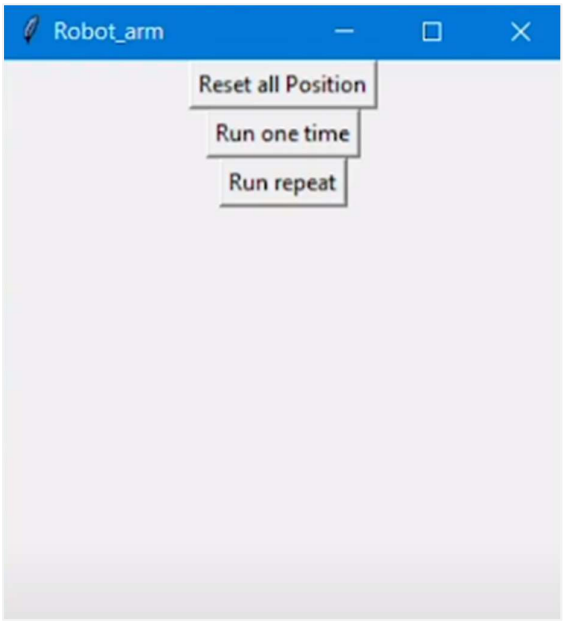
รูปชุดหุ่นยนต์เมื่อต่อเข้ากับไมโครคอนโทรลเลอร์แล้ว

ในตอนที่ 5 นี้คุณจะได้ทดลองสร้างระบบสั่งการหุ่นยนต์แขนกล ให้เริ่มการทำงานตามที่ตั้งค่าไว้ โดยแบ่งรูปแบบการสั่งการออกเป็น 2 แบบ ได้แก่ 1) สั่งการให้หุ่นยนต์ทำงานจนครบกระบวนท่าเพียงรอบเดียว 2) สั่งการให้หุ่นยนต์ทำงานไปแบบวนซ้ำ โดยหลังจากที่คุณติดตั้งหุ่นยนต์เข้ากับไมโครคอนโทรลเลอร์เรียบร้อยแล้ว ในตอนนี้ให้คุณคัดลอกโปรแกรมข้างล่างแล้วนำไปรันใน IDE

```
from tkinter import*
window = Tk()
window.geometry("300x300")
window.title("Robot_arm")
from pyfirmata import Arduino
import time
board = Arduino('COM…')

Servo_A = board.get_pin('d:11:s')
Servo_B = board.get_pin('d:10:s')
Servo_C = board.get_pin('d:9:s')
Servo_D = board.get_pin('d:8:s')
Servo_A.write(80) #limit 70-180
Servo_B.write(90) #limit 50-120
Servo_C.write(120) #limit 90-150
Servo_D.write(90) #limit 5-175
time.sleep(0.5)
print("Communication Successfully")

def reset_position():
        Servo_A.write(80)
        Servo_B.write(90)
        Servo_C.write(120)
        Servo_D.write(90)
        time.sleep(0.5)

def arm_action():
        Servo_A.write(90)
        time.sleep(0.5)
        Servo_B.write(90)
        time.sleep(0.5)
        Servo_C.write(150)
        time.sleep(0.5)
        Servo_D.write(90)
        time.sleep(0.5)

        Servo_A.write(180)
        time.sleep(0.5)
        Servo_B.write(120)
        time.sleep(0.5)
        Servo_C.write(125)
        time.sleep(0.5)
        Servo_D.write(90)
        time.sleep(0.5)
```

```
        Servo_A.write(180)
        time.sleep(0.5)
        Servo_B.write(80)
        time.sleep(0.5)
        Servo_C.write(150)
        time.sleep(0.5)
        Servo_D.write(90)
        time.sleep(0.5)
        reset_position()

def run_onece():
        arm_action()

def run_repeat():
        while(1):
                arm_action()

B0 = Button(window,text = "Reset all Position",
command=reset_position)
B0.pack()
B1 = Button(window, text="Run one time",command = run_onece)
B1.pack()
B2 = Button(window,text="Run repeat", command = run_repeat)
B2.pack()
```

ผลลัพธ์



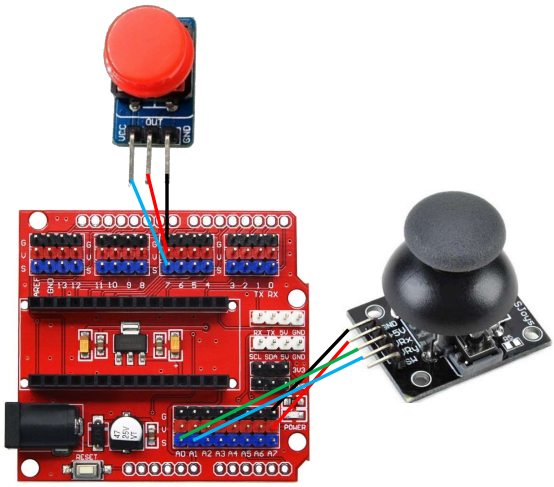
ตอนที่ 6 การรับค่าจากเซนเซอร์

เซนเซอร์ที่ใช้ในการทำหุ่นยนต์ โดยทั่วไปแล้ว สามารถแบ่งออกเป็น 2 ชนิด ได้แก่

1) เซนเซอร์ชนิดดิจิทัล (Digital Sensor) เป็นเซนเซอร์ที่จะให้ผลลัพธ์ (Output) ในรูปแบบสัญญาณที่ไม่ต่อเนื่อง มีสถานะอยู่ เพียง 2 สถานะคือ 0 และ 1 เท่านั้น ทั้งนี้เซนเซอร์ชนิดดิจิทัลที่สามารถนำมาใช้อธิบายหลักการได้ดีที่สุดคือ เซนเซอร์ปุ่มกด ซึ่งมีสถานะเพียง 2 สถานะ คือ ปล่อย และ กด

2) เซนเซอร์ชนิดอนาล็อก (Analog Sensor) เป็นเซ็นเซอร์ที่จะให้ผลลัพธ์ (Output) ในรูปแบบของสัญญาณ หรือ แรงดันที่เปลี่ยนแปลงไปตามปริมาณของการวัด ซึ่งเป็นปริมาณที่มีความต่อเนื่อง ยกตัวอย่างเซนเซอร์ที่มีปริมาณที่เป็นเชิงเส้นต่อเนื่อง ได้แก่ เซนเซอร์วัดการสะท้อนแสง (มืด สลัวๆ สว่าง สว่างจ้า) เซนเซอร์วัดความเข้มเสียง (เงียบ เบา ดัง ดังมาก) และ เซนเซอร์วัดอุณหภูมิ (เย็นจัด เย็น อุ่น ร้อน) เป็นต้น

ตัวอย่างการใช้เซนเซอร์ในหลักสูตรนี้ จะเป็นการใช้โมดูลคันโยก(Joy Stick) เพื่อทดลองการอ่านค่าอนาล็อก และ โมดูลปุ่มกดเพื่อทดลองการอ่านค่าดิจิทัล ตอนนี้ให้คุณเชื่อมต่อสายไฟระหว่างโมดูลคันโยก และ โมดูลปุ่มกด กับไมโครคอนโทรลเลอร์ดังภาพ แล้วคัดลอกโปรแกรมข้างล่างแล้วนำไปรันใน IDE เพื่อทดลองอ่านค่าจากเซนเซอร์ของโมดูลดังกล่าว



รูปแสดงการต่อเซนเซอร์ Digital และ Analog เข้ากับบอร์ดขยาย Arduino Nano

```
from pyfirmata import*
import time
board = Arduino('COM…')
it = util.Iterator(board)
it.start()
print("Communication Successfully")

Sensor_x = board.get_pin('a:0:i')
Sensor_y = board.get_pin('a:1:i')
Sensor_sw = board.get_pin('d:7:i')

while(True):
    x=Sensor_x.read()
    y=Sensor_y.read()
    sw=Sensor_sw.read()
    print("x= ",end=" ");print(x,end=" ")
    print("y= ",end=" ");print(y,end=" ")
    print("sw= ",end=" ");print(sw)
    time.sleep(0.1)
```

ผลลัพธ์

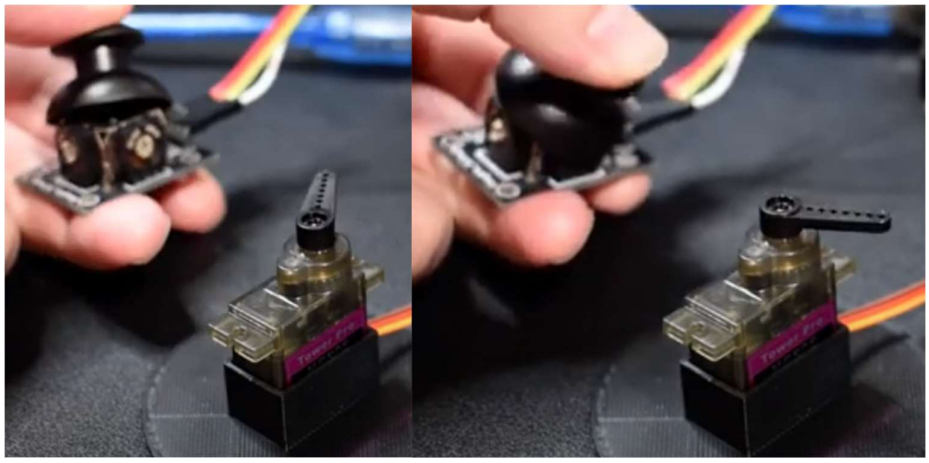
```
x=  0.4897 y=  0.5083 sw=  True
x=  0.4897 y=  0.5064 sw=  True
x=  0.4907 y=  0.5064 sw=  True
x=  0.4888 y=  0.5064 sw=  True
x=  0.4907 y=  0.5073 sw=  True
x=  0.4907 y=  0.5073 sw=  True
```

ตอนที่ 7 การควบคุม Servo 180 degree ด้วยโมดูลคันโยก

ในตอนที่ 7 นี้คุณจะได้ทดลองควบคุมตำแหน่งองศาของ Servo 180 degree ด้วยการใช้ค่าที่อ่านได้จากโมดูลเซนเซอร์ชนิดคันโยก ในตอนนี้ให้คุณคัดลอกโปรแกรมข้างล่างแล้วนำไปรันใน IDE

```
from pyfirmata import*
import time
board = Arduino('COM…')

it = util.Iterator(board)
it.start()

print("Communication Successfully")

Sensor_x = board.get_pin('a:0:i')
Servo3 = board.get_pin('d:3:s')

while(True):
    x=Sensor_x.read()
    if (x == None):
       pass
    else:
       print("x= ",end=" ");print(x)
       Servo3.write(x*180)
       time.sleep(0.001)
```

ผลลัพธ์



ประวัติผู้แต่ง

ร้อยโท ดร.ชัชรินทร์ เลิศยศบดินทร์

ตำแหน่งหน้าที่

2560 - ปัจจุบัน อาจารย์ผู้ช่วยประจำกองวิชาวิทยาศาสตร์
โรงเรียนเตรียมทหารสถาบันวิชาการป้องกันประเทศ

2562 - ปัจจุบัน ผู้ร่วมก่อตั้ง และ ฝ่ายวิชาการ โครงการการเรียนรู้นวัตกรรมและการศึกษาเชิงประลอง “Innovedex”

2558 – 2560 อาจารย์ประจำวิชางานช่าง โรงเรียนสาธิตจุฬาลงกรณ์มหาวิทยาลัย ฝ่ายมัธยม
คณะครุศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย

การศึกษา

พ.ศ. 2566 ปรัชญาดุษฎีบัณฑิต สาขานวัตกรรมการเรียนรู้และเทคโนโลยี
คณะครุศาสตร์อุตสาหกรรมและเทคโนโลยี
มหาวิทยาลัยเทคโนโลยีพระจอมเกล้าธนบุรี

พ.ศ. 2565 หลักสูตรประกาศนียบัตร MIT Professional Education "Leadership & Innovation" Massachusetts Institute of Technology สหรัฐอเมริกา

พ.ศ. 2565 หลักสูตรประกาศนียบัตร Asian Affairs Center "Global Leadership" University of Missouri สหรัฐอเมริกา

พ.ศ. 2563 ครุศาสตร์อุตสาหกรรมมหาบัณฑิต สาขาเทคโนโลยีการเรียนรู้และสื่อสารมวลชน
คณะครุศาสตร์อุตสาหกรรมและเทคโนโลยี
มหาวิทยาลัยเทคโนโลยีพระจอมเกล้าธนบุรี

พ.ศ. 2561 หลักสูตรฝึกอบรม "Global Problem-Based Learning" Shibaura Institute of Technology ณ ประเทศญี่ปุ่น

พ.ศ. 2558 เทคโนโลยีบัณฑิต สาขาเทคโนโลยีการศึกษาและสื่อสารมวลชน
คณะครุศาสตร์อุตสาหกรรมและเทคโนโลยี
มหาวิทยาลัยเทคโนโลยีพระจอมเกล้าธนบุรี

บทความวิชาการ

1. ชัชรินทร์ เลิศยศบดินทร์. (2566). "BASIC" ทักษะพื้นฐานทางดิจิทัลสำหรับข้าราชการทหารสายวิทยาการการศึกษา, วารสารสถาบันวิชาการป้องกันประเทศ. ปีที่14, ฉบับที่1.
2. Chacharin Lertyosbordin, Nichaput Khurukitwanit, Teeratas Asavareongchai, Sirin Liukasemsarn. (2023). Making Music More Inclusive with Hospiano, HRI '23: Companion of the 2023 ACM/IEEE International Conference on Human-Robot Interaction. 13-16 March 2023. pp. 795–797.
3. Chacharin Lertyosbordin, Sorakrich Maneewan, Matt Easter. (2022). Components and Indicators of the Robot Programming Skill Assessment Based on Higher Order Thinking. Applied System Innovation. Volume 5, No. 3, p.47
4. Chacharin Lertyosbordin, Sorakrich Maneewan, Daruwan Srikaew. (2021). Components and Indicators of Problem-solving Skills in Robot Programming Activities. International Journal of Advanced Computer Science and Applications. Volume 10, No. 2, pp.18-22.
5. Chacharin Lertyosbordin, Sorakrich Maneewan, Surapon Boonlue. (2021). Conceptual framework of Google's Online Learning Tools for Python Programming Activity on Challenge-Based Learning. Turkish Online Journal of Qualitative Inquiry. Volume 12, No. 6.
6. ชัชรินทร์ เลิศยศบดินทร์, ปรัชญา ปารมี, และ ณรงค์ฤทธิ์ นาคเม้า. (2563). การออกแบบกิจกรรมการเรียนรู้ทางวิทยาการคำนวณ โดยใช้กระบวนการออกแบบเชิงวิศวกรรม ร่วมกับการจำลองสถานการณ์ออนไลน์ เพื่อส่งเสริมความสามารถในการแก้ปัญหา สำหรับนักเรียนเตรียมทหาร. วารสารครุสาสตร์. ปีที่ 48, ฉบับที่ 2.
7. Chacharin Lertyosbordin, Sorakrich Maneewan, Sakesun Yampinij, Kuntida Thamwipat. (2019). Scoring Rubric of Problem-Solving on Computing Science Learning. International Education Studies. Volume 12, No. 8, pp. 26-32
8. Chacharin Lertyosbordin, Sorakrich Maneewan, Vitsanu Nittayathammakul. (2018). Development of training model on robot programming to enhance creative problem–solving and collaborative learning for mathematics–science program students. Interdisciplinary Research Review. Volume 13, No. 1, pp.61-66

รางวัลประกวดนวัตกรรม

1. รางวัล Best Design in Physical Robot Category การแข่งขัน Student Design Competition ในงาน ACM/IEEE International Conference on Human-Robot Interaction วันที่ 13-16 มีนาคม 2566 เมืองสต็อกโฮล์ม ประเทศสวีเดน
2. รางวัลเหรียญทองแดง ผลงาน "หุ่นยนต์โขนนวัตกรรมสำหรับสตีมศึกษา" งานประกวดนวัตกรรมระดับนานาชาติ 47th International Exhibition of Inventions Geneva วันที่ 11 เมษายน 2562 เมืองเจนีวา สมาพันธรัฐสวิสเซอร์แลนด์
3. รางวัลเหรียญทองแดง ผลงาน "หุ่นยนต์ช่วยสอนเพื่อส่งเสริมความผูกพันและการเรียนรู้" งานประกวดนวัตกรรมระดับนานาชาติ 46th International Exhibition of Inventions Geneva วันที่ 13 เมษายน 2561 เมืองเจนีวา สมาพันธรัฐสวิสเซอร์แลนด์

รางวัลการแข่งขันหุ่นยต์

1. รางวัลผู้ควบคุมทีมดีเด่น การแข่งขันโอลิมปิกหุ่นยนต์ระดับชาติ (WRO) ประเภทความคิดสร้างสรรค์ ณ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง วันที่ 6-8 กันยายน 2562
2. รางวัลผู้ควบคุมทีม รางวัลรองชนะเลิศ การแข่งขันโอลิมปิกหุ่นยนต์ระดับนานาชาติ (Hongkong-IRO) ประเภทหุ่นยต์เตะจุดโทษ ณ เขตบริหารพิเศษฮ่องกงแห่งสาธารณรัฐประชาชนจีน วันที่ 1-6 สิงหาคม 2559
3. รางวัลเหรียญทอง Robot Soccer รุ่นอายุไม่เกิน 19 ปี การแข่งขันโอลิมปิกหุ่นยนต์ระดับนานาชาติ (WRO) ณ เมืองอาบูดาบี สหรัฐอาหรับเอมิเรตส์ วันที่ 18-20 พฤศจิกายน 2554