

# การใช้ While Loop ใน Python

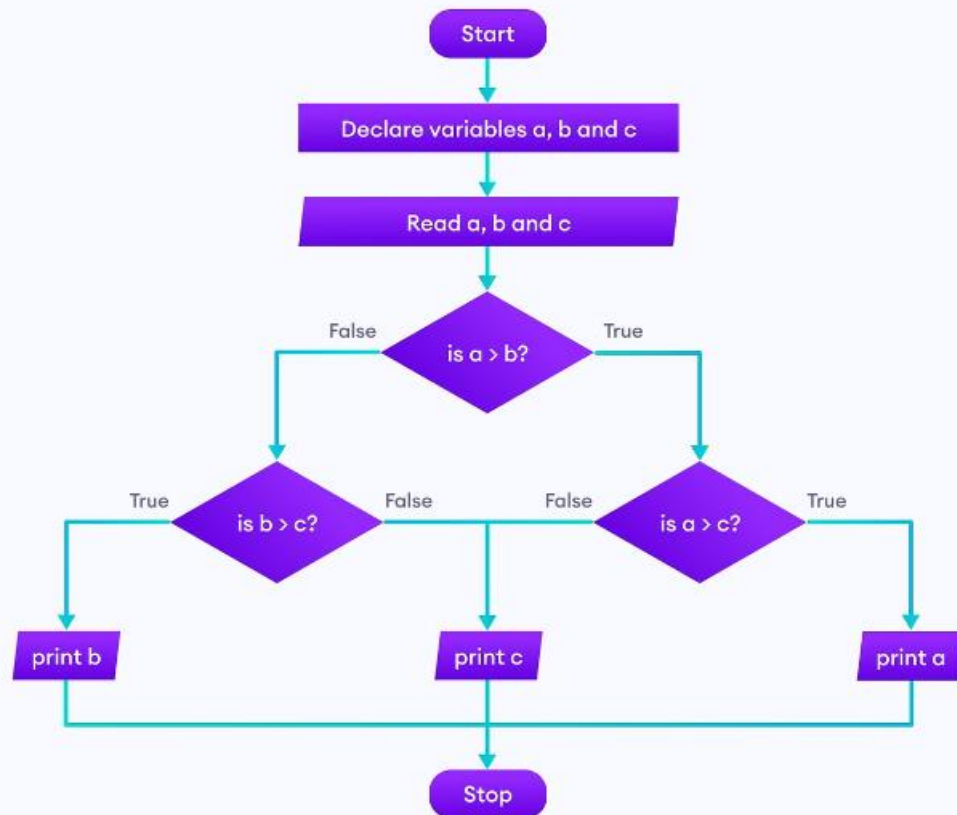
Shutchon Premchaisawatt

```
62 <!--<h3 id="fotovirs">? if ($_COOKIE['lang'] == 'eng'){
63     echo "Photo gallery";
64     elseif ($_COOKIE['lang'] == 'rus') {
65         echo "Фотогалерея";
66     }
67     else
68         echo "Foto galerija";
69     ?></h3>-->
70
71 <div class="<?if($_GET[type]==1)!!$_GET[type])echo "current";>
72 <a href="foto-galerija.php?type=1&text_margin">
73     <div id="left_sidebar">
74         <div id="left_ico"> </div>
75         <p <?if($_COOKIE['lang'] == 'rus')echo "style='margin-left: 50px;";
76     <?
77     if($_COOKIE['lang'] == 'eng'){
78         echo "Wood-frame houses";
79     }elseif($_COOKIE['lang'] == 'rus'){
80         echo "Деревянные каркасные дома";
81     }else{
82         echo "Koka karkasa mājās";
83     }
```

ในโลกของการเขียนโปรแกรม การใช้ลูป (Loop) เป็นสิ่งที่จำเป็นอย่างยิ่งในการสร้างโปรแกรมที่มีประสิทธิภาพ ลูปช่วยให้เราสามารถทำซ้ำชุดคำสั่งได้อย่างง่ายดาย โดยไม่ต้องเขียนรหัสซ้ำๆ หลายครั้ง While Loop เป็นหนึ่งในประเภทของลูปที่ได้รับความนิยมอย่างมากในภาษา Python ด้วยโครงสร้างที่เรียบง่ายและการควบคุมการทำงานที่ยืดหยุ่น While Loop ทำให้การเขียนโปรแกรมซับซ้อนกลายเป็นเรื่องง่าย

# ความหมายของ While Loop

While Loop เป็นโครงสร้างการควบคุมการทำงานในภาษา Python ที่จะทำซ้ำชุดคำสั่งภายในลูปตราบใดที่เงื่อนไขที่กำหนดไว้ยังคงเป็นจริง โดยทั่วไป While Loop จะตรวจสอบเงื่อนไขก่อนที่จะเริ่มทำซ้ำชุดคำสั่ง ดังนั้นหากเงื่อนไขไม่เป็นจริงตั้งแต่แรก ชุดคำสั่งภายในลูปจะไม่ได้รับการดำเนินการเลย



1

## เงื่อนไข

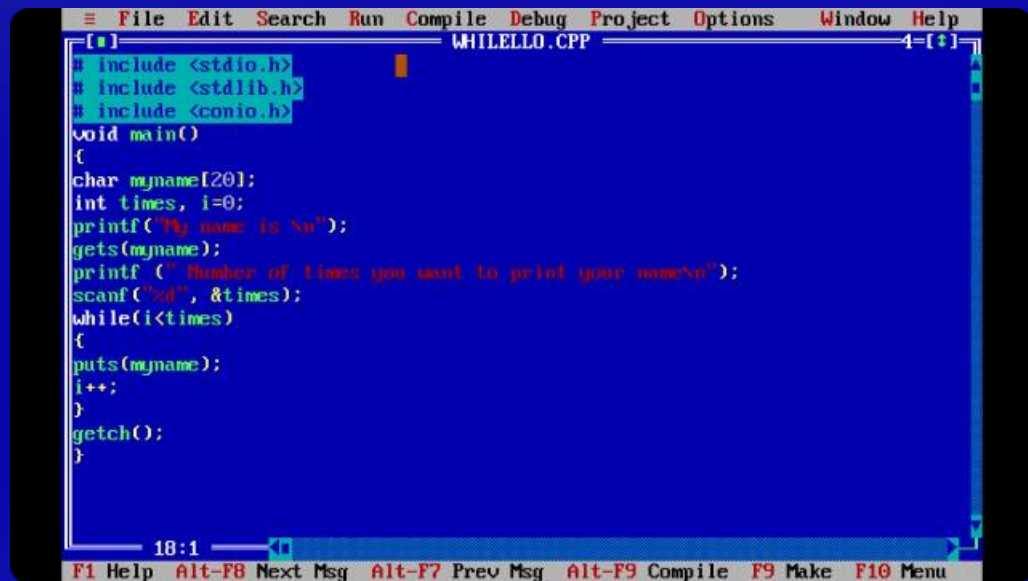
เป็นนิพจน์ที่ส่งผลลัพธ์เป็นค่าความจริง (True หรือ False) โดยเงื่อนไขจะถูกตรวจสอบก่อนที่จะทำซ้ำชุดคำสั่ง

2

## ชุดคำสั่ง

เป็นรหัส Python ที่จะถูกดำเนินการซ้ำๆ ตราบใดที่เงื่อนไขเป็นจริง

# โครงสร้างของ While Loop

A screenshot of a C++ IDE window titled 'WHILELLO.CPP'. The code is as follows:

```
#include <stdio.h>
#include <stdlib.h>
#include <conio.h>
void main()
{
    char myname[20];
    int times, i=0;
    printf("My name is Su");
    gets(myname);
    printf ("Number of times you want to print your names");
    scanf("%d", &times);
    while(i<times)
    {
        puts(myname);
        i++;
    }
    getch();
}
```

The IDE has a menu bar with File, Edit, Search, Run, Compile, Debug, Project, Options, Window, and Help. The status bar at the bottom shows '18:1' and function key shortcuts: F1 Help, Alt-F8 Next Msg, Alt-F7 Prev Msg, Alt-F9 Compile, F9 Make, and F10 Menu.

โครงสร้างของ While Loop ในภาษา Python นั้นเรียบง่ายและง่ายต่อการทำความเข้าใจ โดยทั่วไป While Loop จะมีโครงสร้างดังนี้

while เงื่อนไข:  
    ชุดคำสั่ง

ในโครงสร้างนี้ "เงื่อนไข" เป็นนิพจน์ที่ส่งผลลัพธ์เป็นค่าความจริง (True หรือ False) "ชุดคำสั่ง" เป็นรหัส Python ที่จะถูกดำเนินการซ้ำๆ トラบใดที่ "เงื่อนไข" เป็นจริง

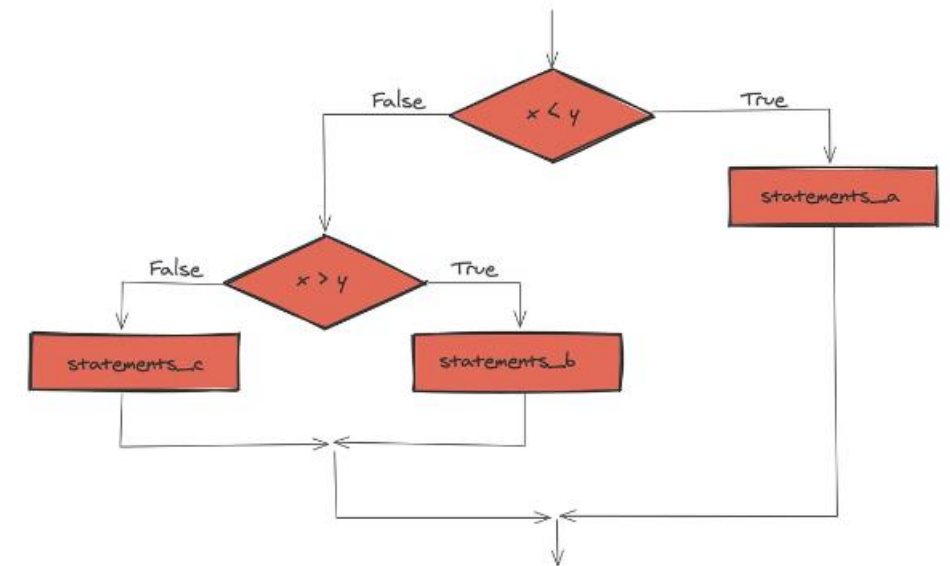
# การใช้เงื่อนไขใน While Loop

การใช้เงื่อนไขใน While Loop เป็นสิ่งที่สำคัญมาก เนื่องจากเงื่อนไขเป็นตัวควบคุมการทำงานของลูป โดยทั่วไป เงื่อนไขจะใช้เพื่อตรวจสอบว่าลูปควรทำซ้ำต่อไปหรือไม่ หากเงื่อนไขเป็นจริง ลูปจะทำซ้ำชุดคำสั่ง และหากเงื่อนไขเป็นเท็จ ลูปจะสิ้นสุด

ตัวอย่างเช่น

```
count = 0
while count < 5:
    print(count)
    count += 1
```

ในตัวอย่างนี้ เงื่อนไขคือ "count < 5" ชุดคำสั่งภายในลูปคือ "print(count)" และ "count += 1" ลูปจะทำซ้ำจนกว่า "count" จะมีค่าเท่ากับ 5



# การใช้ Break และ Continue ใน While While Loop

Break และ Continue เป็นคำสั่งพิเศษที่ใช้ในการควบคุมการทำงานของ While Loop Break ใช้สำหรับหยุดการทำงานของลูปโดยสิ้นเชิง ในขณะที่ Continue ใช้สำหรับข้ามการดำเนินการของชุดคำสั่งปัจจุบันและเริ่มทำซ้ำชุดคำสั่งใหม่

ตัวอย่างเช่น

```
count = 0
while count < 10:
    if count == 5:
        break
    print(count)
    count += 1
```

ในตัวอย่างนี้ ลูปจะทำซ้ำจนกว่า "count" จะมีค่าเท่ากับ 5 เมื่อ "count" มีค่าเท่ากับ 5 คำสั่ง "break" จะถูกเรียกใช้ ทำให้ลูปหยุดทำงาน

## *continue* and *break* statement

### ■ Example:

```
for (a = 0; a < 5; a++)
{
    if (a == 2) continue;
    printf("a = %d\n", a);
}
```

### ■ Output:

```
a = 0
a = 1
a = 3
a = 4
```

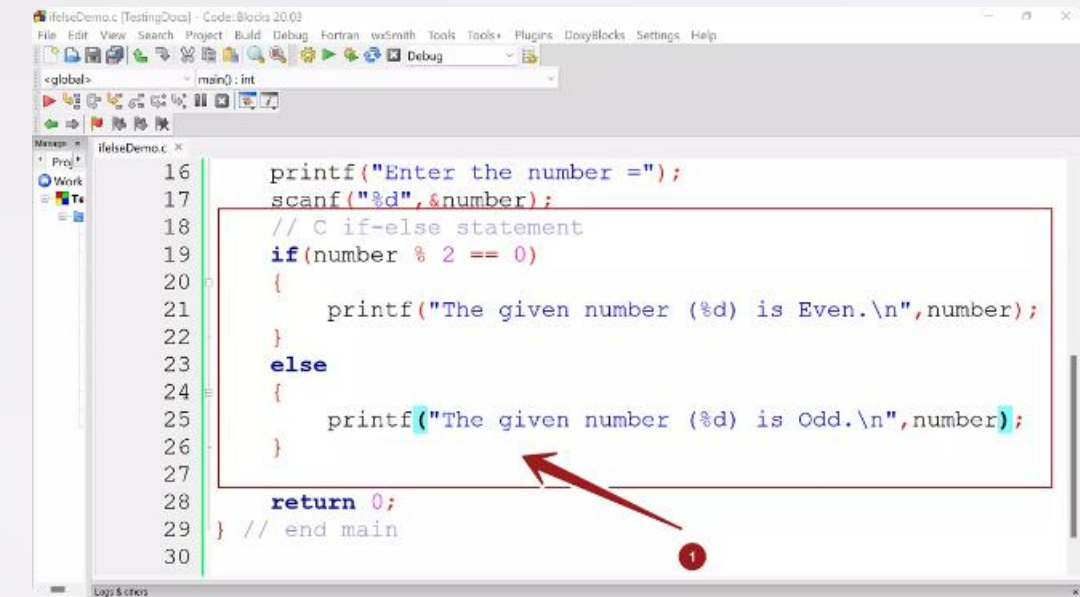
# การใช้ if else ร่วมกับ While Loop

เราสามารถใช้ if else statement ร่วมกับ While Loop เพื่อเพิ่มความยืดหยุ่นในการควบคุมการทำงานของลูป โดยทั่วไป if else statement จะถูกใช้เพื่อตรวจสอบเงื่อนไขเพิ่มเติมภายในลูป และดำเนินการชุดคำสั่งที่แตกต่างกันตามเงื่อนไขเหล่านั้น

ตัวอย่างเช่น

```
count = 0
while count < 10:
    if count % 2 == 0:
        print(count, "เป็นเลขคู่")
    else:
        print(count, "เป็นเลขคี่")
    count += 1
```

ในตัวอย่างนี้ ลูปจะทำซ้ำจนกว่า "count" จะมีค่าเท่ากับ 10 if else statement จะถูกใช้เพื่อตรวจสอบว่า "count" เป็นเลขคู่หรือเลขคี่ และพิมพ์ข้อความที่แตกต่างกันตามเงื่อนไข



```
16 printf("Enter the number =");
17 scanf("%d",&number);
18 // C if-else statement
19 if(number % 2 == 0)
20 {
21     printf("The given number (%d) is Even.\n",number);
22 }
23 else
24 {
25     printf("The given number (%d) is Odd.\n",number);
26 }
27
28 return 0;
29 } // end main
30
```



# ตัวอย่างการใช้ While Loop

While Loop มีประโยชน์มากมายในการแก้ปัญหามากมาย ในการเขียนโปรแกรม ตัวอย่างเช่น เราสามารถใช้ While Loop เพื่อรับข้อมูลจากผู้ใช้จนกว่าผู้ใช้จะป้อนข้อมูลที่ถูกต้อง หรือใช้ While Loop เพื่อทำซ้ำชุดคำสั่งจนกว่าเงื่อนไขบางอย่างจะตรงตามความต้องการ

ตัวอย่างเช่น โปรแกรมรับข้อมูลจากผู้ใช้จนกว่าผู้ใช้จะป้อนคำว่า "ออก"

```
while True:
    input_data = input("กรุณาป้อนข้อความ (ป้อน 'ออก' เพื่อออกจากโปรแกรม): ")
    if input_data == "ออก":
        break
    print("คุณป้อนข้อความ:", input_data)
```



# ข้อดีและข้อเสียของ While Loop

While Loop เป็นเครื่องมือที่ทรงพลัง แต่ก็มีข้อดีและข้อเสียเช่นกัน

## ข้อดี

- ควบคุมการทำงานได้อย่างยืดหยุ่น
- เหมาะสำหรับกรณีที่ไม่รู้จำนวนรอบที่แน่นอน
- ง่ายต่อการใช้งานและเข้าใจ

## ข้อเสีย

- อาจเกิดลูปอนันต์หากเงื่อนไขไม่เป็นเท็จ
- อาจเขียนรหัสซับซ้อนได้หากใช้ไม่เหมาะสม



# การใช้ While Loop ในการแก้ปัญหาต่างๆ

While Loop สามารถนำไปใช้ในการแก้ปัญหาต่างๆ ในการเขียนโปรแกรม ตัวอย่างเช่น

- การรับข้อมูลจากผู้ใช้น้อยกว่าผู้ที่จะป้อนข้อมูลที่ถูกต้อง
- การสร้างเมนูแบบอินเตอร์แอคทีฟ
- การทำซ้ำชุดคำสั่งจนกว่าเงื่อนไขบางอย่างจะตรงตามความต้องการ
- การสร้างเกมง่ายๆ
- การประมวลผลข้อมูลจากไฟล์

โดยทั่วไป While Loop เป็นเครื่องมือที่ทรงพลังและสามารถใช้ได้ในหลายๆ บริบท

# สรุปและบทสรุป

While Loop เป็นโครงสร้างการควบคุมการทำงานที่สำคัญในภาษา Python ด้วยโครงสร้างที่เรียบง่ายและความยืดหยุ่นในการควบคุมการทำงาน ทำให้ While Loop เหมาะสำหรับการแก้ปัญหาต่างๆ ในการเขียนโปรแกรม

การทำความเข้าใจแนวคิดของ While Loop จะช่วยให้เราสามารถเขียนโปรแกรมที่มีประสิทธิภาพและสร้างสรรค์ได้มากขึ้น โดยเฉพาะอย่างยิ่งในกรณีที่เรารู้จำนวนรอบที่แน่นอนของการทำซ้ำ

