9 條件分支

9-1. 單向選擇

9-1-1 選擇敘述

在序列結構時，程式是一行接一行、由上往下地循序執行。如果有些敘述想跳過去不執行的話，就要透過選擇結構的敘述來完成。好比程式會轉彎，避開不要執行的敘述，執行需要對應處理的敘述。

選擇敘述表示在不同的條件下，做不同的事情。

- 如果判斷條件成立，就執行條件成立的動作；
- 如果判斷條件不成立，則執行條件不成立的動作。

9-1-2 基本觀念

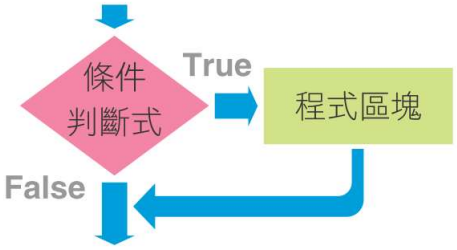
1. 單向選擇結構是最簡單的選擇結構，只有一個方向的選擇，如果條件判斷成立，就執行條件為真的動作。

例如：「若週末天氣好的話，我們就去打球」。

圖5-1　單向選擇敘述示意圖

2. 語法: 只有if指令

單向判斷（if）

Python 語言	流程圖
`if 條件判斷:` `　　[程式區塊]`	
說明：如果條件判斷成立或為 True，就執行「程式區塊」，若不成立則不會執行程式區塊。	

```
if <布林條件式>:
    <當條件式為True時，所要執行的程式碼。>
```

要點1: if是條件指令。

要點2: 布林條件式。條件判斷的結果為布林值True或False。

要點3: 冒號:很重要。分隔程式碼區塊。

要點4: 縮排程式碼：表示要執行的程式碼區塊。

=> 在條件敘述式輸入冒號：，後面的每一行都要縮排(indentation)，表示要執行的程式碼區塊。

Python對縮排很挑剔；屬於區塊的每一行程式碼都要有相同的縮排距離，習慣上是四個空白鍵。不要和tab鍵混用，特別是拷貝貼時，容易出錯

範例1: 判斷成績及格

寫一個程式判斷輸入的成績是否，若是及格則顯示「很好，請繼續保持下去」。

解題想法

可以使用單向選擇結構撰寫程式，判斷成績是否及格，及格就顯示「很好，請繼續保持下去」。

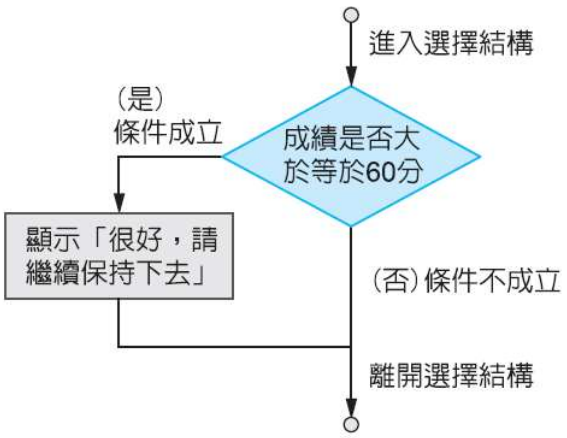


圖 A-2　流程圖

```
score = int(input('請輸入一個成績？'))
if score >= 60:
    print('很好，請保持下去')
```

【隨堂練習1】: 是否加收服務費

若消費金額未滿1200元，加收服務費1成。

```
#|echo: true
#|eval: false
#|includes: false

money = int(input("請輸入消費金額： "))
if money < 1200:
    money = money * 1.1
print(f"需繳付的實際金額為{money}元")
```

9-2. 雙向選擇

1. 雙向選擇結構有兩個方向的選擇：二選一，「如果...就...，否則就...」。

- 如果條件判斷成立，就執行條件為真的動作；
- 如果條件判斷不成立，就執行條件為假的動作。

例如：「如果週末天氣好的話，我們就出去打球，否則就去看電影」。

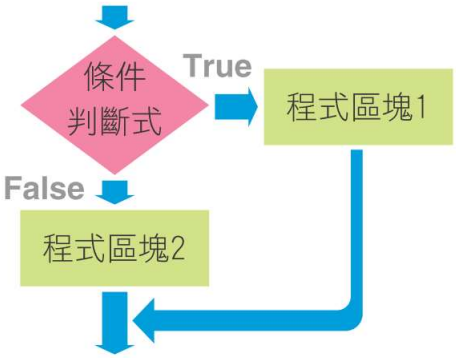
圖5-5　雙向選擇敘述示意圖

2. 語法: if/else指令

布林條件式，有兩個冒號，兩個程式區塊，分屬if區塊和else區塊。

雙向判斷（if ～ else）

Python 語言	流程圖

`if 條件判斷:`
`    [程式區塊1]`
`else:`
`    [程式區塊2]`
	

說明：如果條件判斷成立或為 True，就執行底下「程式區塊 1」，若條件不成立則執行「程式區塊 2」。	

要點1: 雙向選擇用 if...else指令。

要點2: 一個布林條件式：條件判斷的結果會有布林值為True與False的兩種可性。

要點3: 兩個冒號：分隔if與else兩個程式區塊，布林值為真執行if區塊，布林值為假執行else區塊。

要點4: 縮排程式碼：表示要執行的if或else程式碼區塊。

- Python程式區塊，用縮排來表示，而非以一對大括號「{}」表示執行的範圍。
- 同一個程式區塊的程式，每行都要空相同長度，通常是4 個空白鍵(space)。
- Tab 也可用於表示縮排，但是空白鍵與Tab 鍵不要混用。

範例2: 判斷及格與不及格

寫一個程式判斷並顯示所輸入的成績是及格還是不及格。

解題想法

可以使用雙向選擇結構撰寫程式，判斷成績是否及格，及格就顯示「有及格呦～」，不及格就顯示「不及格ㄟ！」。

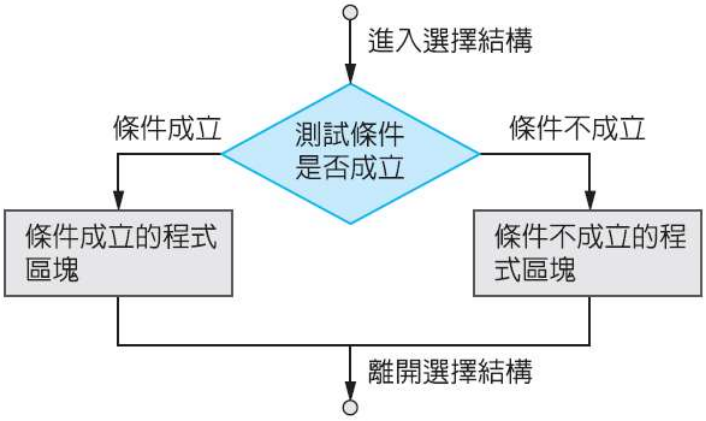


圖 A-3　流程圖

```python
score = int(input('請輸入一個成績？'))
if score >= 60:
    print('有及格ㄡ～')
else:
    print('不及格ㄟ！')

## 簡潔的表達
print("及格") if score >= 60 else print("不及格")
```

【隨堂練習2】: 滿2000 打九折

請寫一個程式幫助店家計算顧客所需付出的金額。

採買物品時，有時會遇到店家為了刺激消費，會使用滿額折扣。例如，滿2000 打九折，未滿2000 則不打折。

解題想法 可以使用雙向選擇結構撰寫程式，判斷購買金額是否在2000 元以上，若購買金額在2000 元以上，輸出購買金額乘以0.9；否則依照原價輸出。

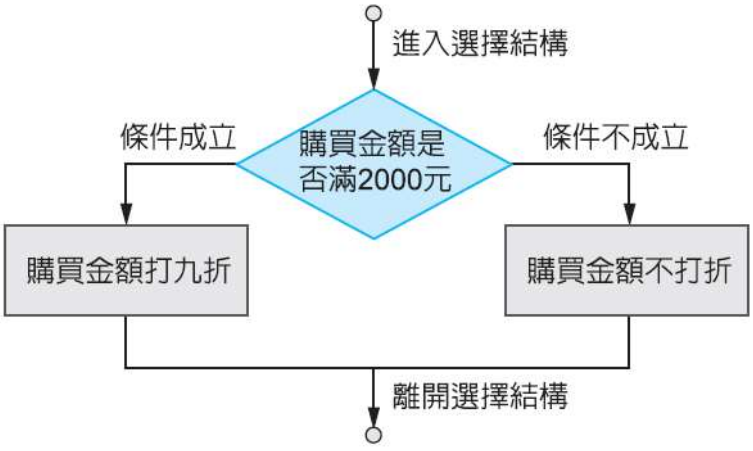


圖 A-4　流程圖

```
#|echo: true
#|eval: false
#|includes: false

cost = int(input("Enter the spending: "))
if cost >= 2000:
    print(cost*0.9)
else:
    print(cost)
```

【隨堂練習3】: 判斷奇偶數

請寫一個程式判斷輸入的值是奇數還是偶數。

解題想法

可以使用雙向選擇結構撰寫程式，判斷輸入值除以2 的餘數，若餘數不為0，則輸出該數為奇數；否則輸出該數為偶數。

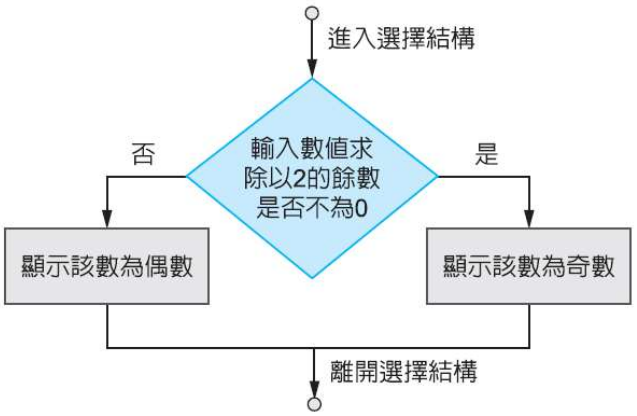


圖 A-5　流程圖

```
#|echo: false
#|eval: false
#|includes: false

num = int(input("Enter an integer: "))
## Process
if num%2 == 0:
    print(num, "為偶數")
else:
    print(num, "為奇數")
```

9-3. 多向選擇

1. 一個問題需要在不同情況做出不同的選擇，就需要使用多向選擇。

例如：「如果週末天氣好的話，我們就出去打球；如果週末陰天的話，我們就在附近公園逛逛；否則就去看電影」。

● 圖5-7　多向選擇敘述示意圖

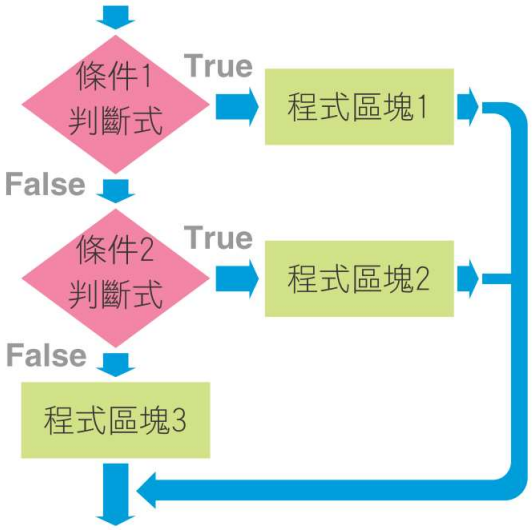
2. 多向選擇結構有多個條件判斷的選擇結果。

- 通常條件最嚴格的放在最上面，其他條件依序放置。
- 由上而下依序進行條件判斷，如果條件判斷成立，就執行該條件為真的動作；如果條件判斷不成立，就繼續執行下一個條件判斷，直到結束 。

3. 語法: if+elif+else指令

多向判斷（if ～ elif ～ else）

Python 語言	流程圖

`if 條件1 判斷:`
`　　[程式區塊1]`
`elif 條件2 判斷:`
`　　[程式區塊2]`
`else:`
`　　[程式區塊3]`
	

說明：如果條件 1 判斷成立或為 True 時，就執行「程式區塊 1」內容，否則再判斷條件 2，當條件 1 不成立且條件 2 成立就執行「程式區塊 2」，當條件 1、2 都不成立才執行「程式區塊 3」的內容。

要點1: 多向選擇用 if...elif...else指令。

要點2: n-1個布林條件式：每個條件判斷的結果會有布林值的True與False兩種可性。

要點3: n個冒號：分隔if, elif與else程式區塊。

要點4: 縮排程式碼：表示要執行的if, elif與else程式碼區塊。

範例3：分數與評語

寫一個程式若成績大於等於80 分，評語為「非常好」，否則若成績大於等於60 分，評語為「不錯喔」，否則評語為「要加油」，將以上敘述表示為表格，如下。

表 4-4　分數與評語

成績	評語
成績 >=80	非常好
80> 成績 >=60	不錯喔
成績 <60	要加油

解題想法

可以使用多向選擇結構撰寫程式，若成績是否大於等於80，則顯示「非常好」，否則若成績大於等於60，則顯示「不錯喔」，否則顯示「要加油」。

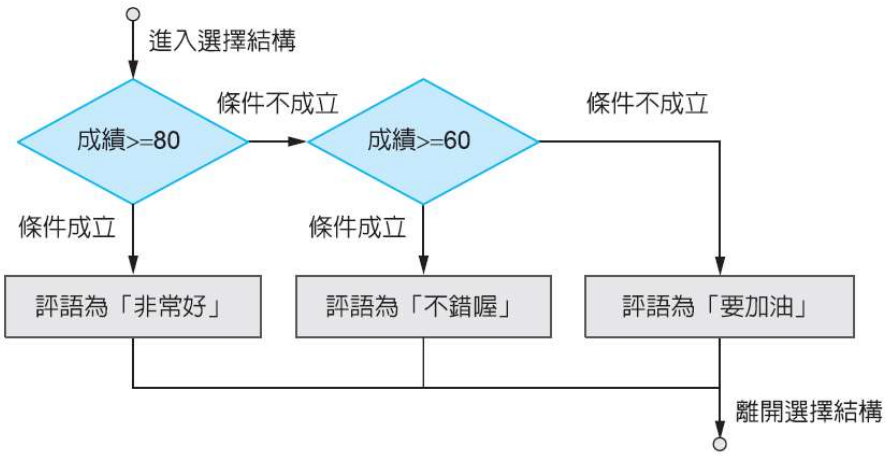


圖 A-8　流程圖

```
score = int(input(' 請輸入一個成績？ '))
if score >= 80:
    print('非常好')
elif score >= 60:
    print(' 不錯喔')
else:
    print(' 要加油')
```

隨堂練習4：閏年判斷

設計程式允許輸入西元幾年，請求出該年是否是閏年，閏年表示該年多一天，若為4的倍數稱做閏年，但若為100的倍數就不為閏年，且若為400倍數又是閏年。(四年一潤、百年不潤、四百年再潤)

輸入年份「2012」，輸出為「2012是閏年」。

～程式執行結果如下。

請輸入年份？2012

2012 是閏年

```
#|echo: true
#|eval: false
#|includes: false

## Input Data
year = int(input("Enter year? "))

## Process Data
if year%400==0:
    print(year, "是潤年")
elif year%100==0:
    print(year, "不是潤年")
elif year%== 0:
    print(year, "是潤年")
else:
    print(year, "不是潤年")


if((year%400==0)or(year%4==0)):
    print(year, "是潤年")
else:
    print(year, "不是潤年")
```

【加分練習】:BMI 計算

請寫一個程式讓使用者輸入身高與體重，顯示BMI 值與肥胖程度。 BMI 等於體重（KG）除以身高（M）的平方，而BMI 與肥胖分類標準如下：

表 4-6　BMI 計算

BMI 值	肥胖分級
BMI < 18	體重過輕
18 ≦ BMI < 24	體重正常
24 ≦ BMI < 27	體重過重
27 ≦ BMI	體重肥胖

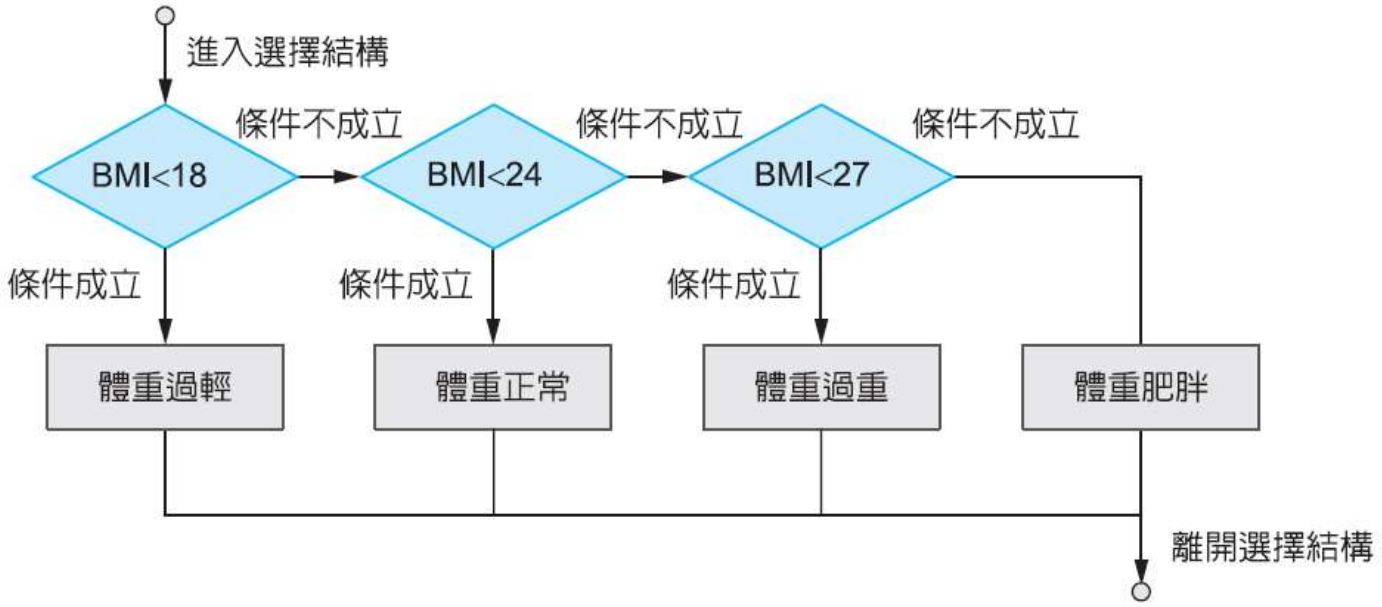


圖 A-10　流程圖

```
#|echo: false
#|eval: false
#|includes: false

## Input Data and Calculate BMI
height = eval(input("Enter height in centimeters: "))
weight = eval(input("Enter weight in kilograms: "))
BMI = weight / (height/100)**2

## Branch
if BMI < 18:
    print("Underwight")
elif BMI < 24:
    print("Normal")
elif BMI < 27:
    print("Overwight")
else:
    print("Obese")
```

9-4. 巢狀選擇

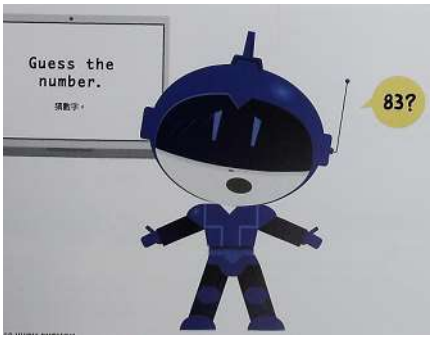
有些事情是在某個條件成立後，才需要再做下一步的條件判斷。也就是說，只有在第一層件式成立時（外層），才會執行第二層條件式（內層），內層條件式屬於外層條件式的一部份。

1. 巢狀選擇(nested choice)為選擇結構內包含選擇結構，是一種有層次的選擇。

- 巢狀選擇結構有好幾層，每一層的選擇都可以使用單向、雙向與多向選擇結構。
- 巢狀結構沒有標準程式語法，每個程式的語法可能都不相同。
- 巢狀選擇結構將會依照程式的需要組合出不同的選擇結果。
- 巢狀選擇結構有無限可能執行的路徑或狀態。

範例4：猜數字遊戲

使用者輸入的數不等於秘密數，則它會看這個數太低或太高，給使用者適當訊息。




```
#|echo: true
#|eval: false
#|includes: false


## 輸入秘密數與猜測數
secret_number = 83
n = int(input("Guess a number between 1 and 100: ")) # 文字轉數字

## 兩個雙向選擇（巢狀選擇）
if n == secret_number:
    print("Guess Right!")
else:     # 猜錯則檢查太高或太低
    if n > secrete_number:
        print("Guess lower!") # 太高要往低猜
    else:
        print("Guess higher") # 太低要往高猜
print("Thank you for joining the game!")
```

範例5: 分數與評語(重要)

寫一個程式若成績大於等於80 分，評語為「非常好」；否則若成績大於等於60 分，評語為「不錯喔」；否則評語為「要加油」。

解題想法

之前我們處理過這一題，用的是多向選擇結構撰寫程式：若成績是否大於等於80，則顯示「非常好」，否則若成績大於等於60，則顯示「不錯喔」，否則顯示「要加油」。

想想看，可不可以用單向選擇、巢狀選擇來處理？程式與結果有什麼不同？

```
## 方法（一）：單向選擇
score = int(input('請輸入一個成績？'))
if score >= 80:
    print('非常好')
if 80 > score >= 60: # 前面要有80>才行
    print('不錯喔')
if score < 60:
    print('要加油')


## 方法（二）：多向選擇
score = int(input('請輸入一個成績？'))

if score >= 80:
    print('非常好')
elif score >= 60:   # 前面不用有80>
    print('不錯喔')
else:
    print('要加油')


```

```python
## 方法（三）：雙向選擇+雙向選擇 => （多個）巢狀選擇
score = int(input('請輸入一個成績？'))
if score >= 80:
    print('非常好')
else:
    if score >= 60:
        print('不錯喔')
    else:
        print('要加油')
```

加分題：字串查詢 - 成員查詢

學校網站上公布了參加元旦聯歡晚會的入選演員名單，參加海選的同學都急切地想知道自己是否被選中。用Python編制一個小程序，幫助同學從眾多的名單中查詢自己的入選結果。
in 包含；not in 不包含

```python
#|echo: true
#|eval: false
#|includes: false

## Input Data
show = "元旦晚會演出名單：趙一、錢二、孫三、李四..."
name = input("輸入要查詢的姓名： ")

## Process Data + Output Data => Conditional Choice
if name in show:
    print("恭喜" +name+ "!榜上有名！")
else:
    print("抱歉" +name+ "下次還有機會！")
```

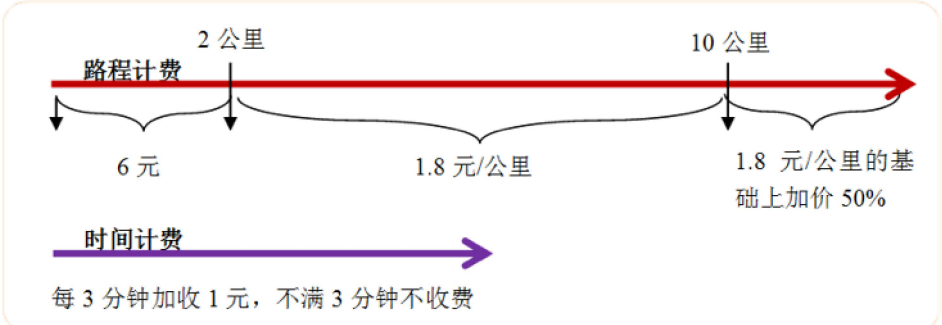
加分題：計程車費

某城市計程車計費3公里以內6元，超過3公里不足10公里、每公里1.8元，超過10公里則超過部分加收50%。此外停車等候每3分鐘收1元，請用Python寫出車費是多少。


研究室

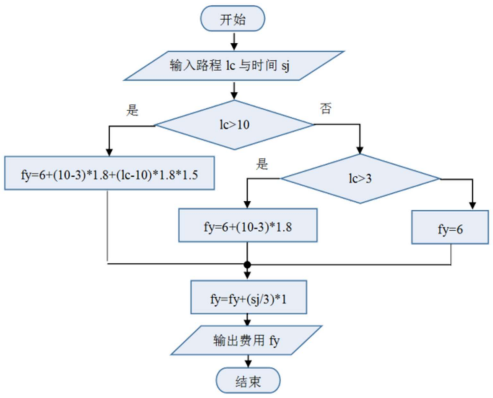
1．思路分析

打车计价方案：3 公里以内起步是 6 元，超过 3 公里之后按 1.8 元/公里计价；超过 10 公里之后在 1.8 元/公里的基础上加价 50%。此外，停车等候则按时间计费：每 3 分钟加收 1 元（注：不满 3 分钟不计费）。



2．算法描述

第一步：输入路程 lc 与时间 sj。
第二步：判断路程，根据不同路程计算费用。
第三步：判断时间，根据时长计算费用。
第四步：统计总费用：路程费用+时长费用。
第五步：输出费用，程序结束。




```
#|echo: true
#|eval: false
#|includes: false

## Input Data - Distance and Time
length = int(input("請輸入距離： "))
time = int(input("請輸入時間： "))

## Process Data => Conditional Choice => Calculate fare
if length < 3:
    money_distance = 6
elif length < 10:
    money_distance = 6 + (length - 3) * 1.8
else:
    money_distance = 6 + (10-3)*1.8+(length-10)*1.8*1.5

## 時間費率
money_time = (time/3) * 1
## 距離與時間加總
feight = money_distance + money_time

## Output Data
print("Total fare is", feight)
```