Introduction to Computer Programming

2.2 User input & nested conditionals

Input

`input` : accepts typed input from the user and outputs the typed input as string data.

The function argument is a string, which is the prompt displayed to the user.

In [1]:

```
name = input("Enter your name: ") # type response when prompted and press enter

print('My name is ', name)
```

```
Enter your name:
My name is
```

This is a quick and easy way to add dynamic input to your program.

Input - a word of warning!

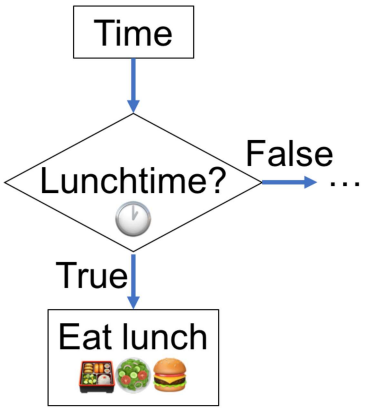
The input by the user is stored as a string.
Numbers entered will behave as text data unless converted to a numerical data type.

In [2]:

```
A = input("Enter a number ")
B = input("Enter another number ")

print(A + B)

print(int(A) + int(B))
```

```
Enter a number
Enter another number


---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-2-23fd8189f69c> in <module>
      4 print(A + B)
      5
----> 6 print(int(A) + int(B))

ValueError: invalid literal for int() with base 10: ''
```

Example: Write a program that requests the time from the user and tells them to eat lunch if it is lunchtime.



In [11]:

```
t = input('enter the time (24 hour clock) in format hh.mm : ')
t = float(t)    # string converted to float


# ----------------- Program from last week ---------------------
Ls = 13.00      # lunch starts
Le = 14.00      # lunch ends

lunchtime = t >= Ls and t < Le

if lunchtime:
    print("Eat lunch")
# ------------------------------------------------------------
```

```
enter the time (24 hour clock) in format hh.mm : 3.00
```

`split` : splits a string into a list.

In [6]:

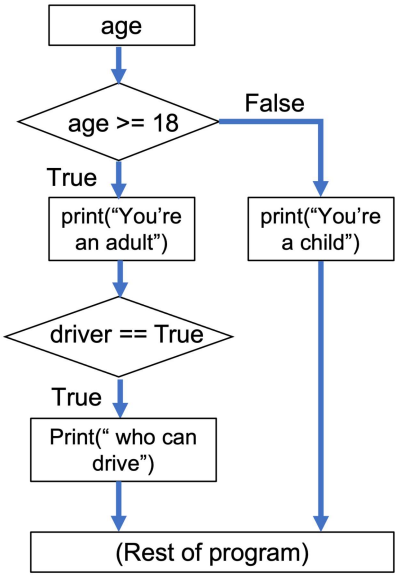
```
A = input("Enter two numbers ")
print(A)

B = A.split()
print(B)
```

```
Enter two numbers 3 4
3 4
['3', '4']
```

Nested conditional statements

Conditional statements can be nested (a conditional statement within a conditional statement) to execute more complex decision making in a program.

Example: Translate the flow diagram into a programme



In [3]:

```
age = 19
graduated = False
driver = True

# Check if person is 18 years or older
if age >= 18:
    print("You're an adult", end='')

    if driver:
        print(' who can drive')
else:
    print("You're a child")
```

```
You're an adult who can drive
```

Summary

- `input` : accepts typed input from the user. Remember that the function outputs the typed input as string data!
- Conditional statements can be nested to create more complex decision-making within a program.

In-class Demos

Example 1: Write a program that asks for the user's age and checks if they are 18 or over.

In [21]:

```
age = input('Enter your age: ')

if int(age) >= 18:
    print('adult')
```

```
Enter your age: 34
adult
```

Example 2: Write a program for an online shop that:

- asks for the number of items ordered
- check how many are in stock and prints a message to re-supply if the order cannot be fulfilled
- if the items are in stock, prints a message if there will be multiple packages shipped assuming 8 items fit in one box.

In [20]:

```
Ordered = int(input('No items ordered: '))
InStock = 32

if InStock < Ordered:
    print('re-supply!')
else:
    if Ordered/8 > 1:
        print('mutiple packages!')
```

```
No items ordered: 56
re-supply!
```

Example 3: Write a program that:

- checks if a number of odd or even.
- checks if input even numbers are multiples of 4.
- checks if input odd numbers are multiples of 3.

In [24]:

```
N = 9

if not N % 2:
    print('Even number', end='')
    if not N % 4:
        print(' and mulitple of 4')
else:
    print('Odd number', end='')
    if not N % 3:
        print(' and multiple of 3')
```

```
Odd number and multiple of 3
```

In [ ]:

```

```