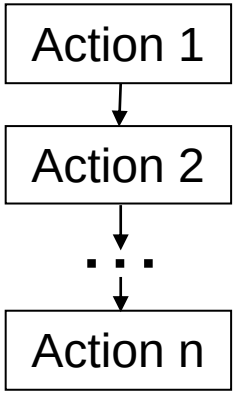
Lesson 3b: Python Conditional control structures

Sequential composition

- The examples of algorithms we have seen so far have been simple instructional sequences.
- It is the easiest structure of all: we tell the processor that it must consecutively execute a list of actions.
- To build a sequence we write one action per line.



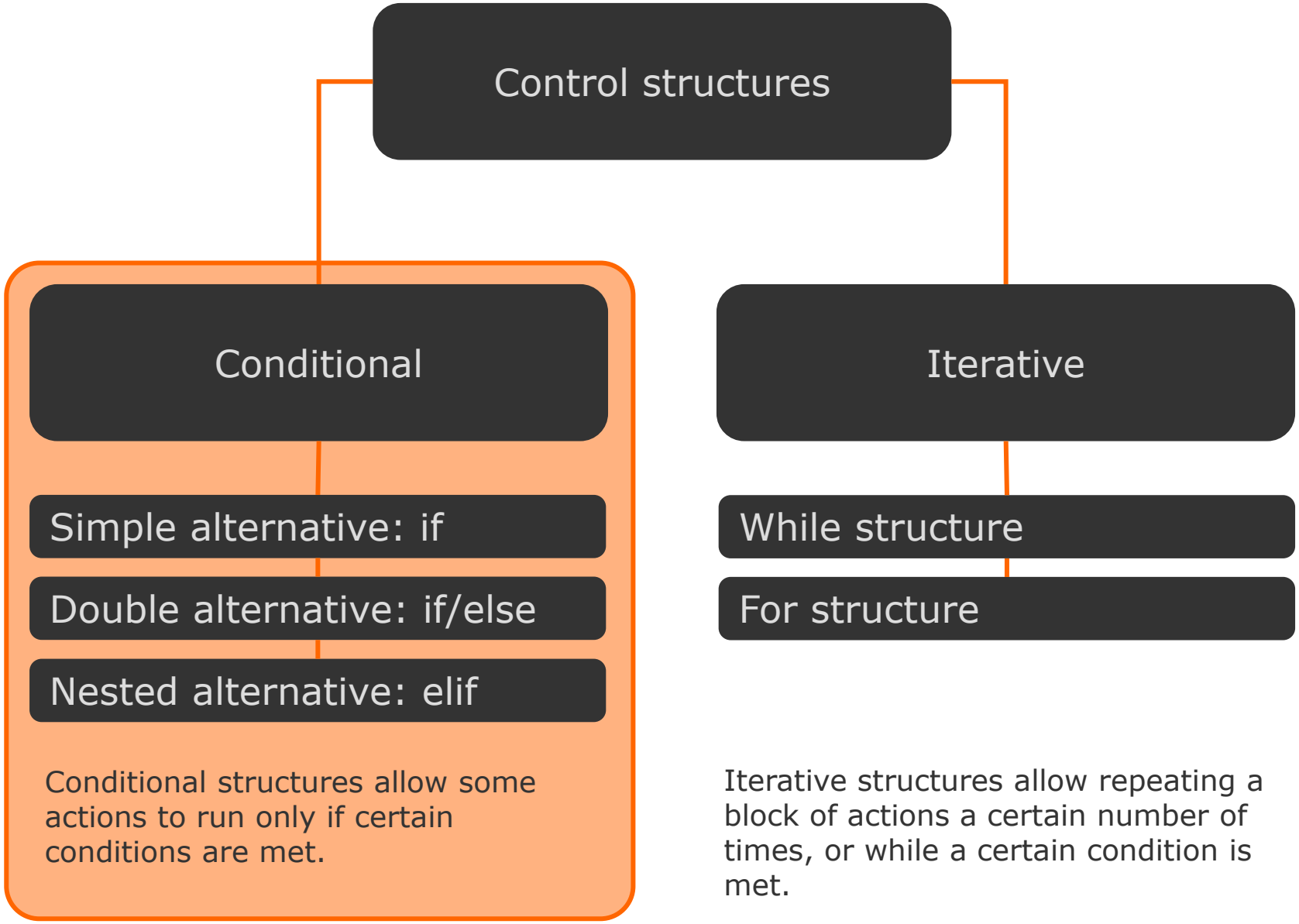
<action 1>
<action 2>
...
<action n>

```
n1 = float(input())
n2 = float(input())
n3 = float(input())
mean_nums = (n1+n2+n3)/3
print(mean_nums)
```

- There is the possibility of expressing a sequential structure by writing several separate actions with , (not readable)

```
n1 = float(input()), n2 = float(input()), n3 = float(input())
mean_nums = (n1+n2+n3)/3, print(mean_nums)
```

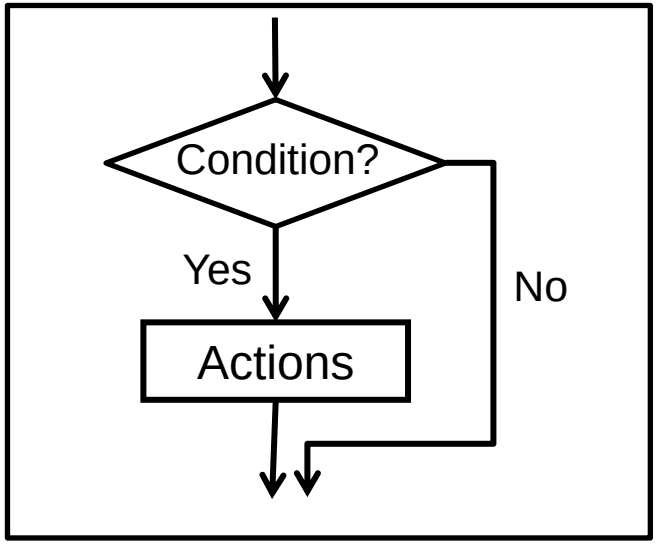
Control structures



Conditional structures: simple alternative

In the simple alternative we have a set of actions that will only be executed if a certain condition is met. If the condition is not met, nothing runs.

Flowchart



Syntax

```
if (<condition>):
  <action 1>
  <action 2>
   …
  <action N>
```

Important:

Indicate start actions with :

Indentation

Exercise

We propose to write the following program:

Ask the user to introduce an integer value. If the value is inside the interval [0,10], display the message:

"Number <write the number> is **NOT** inside the interval"

Otherwise display:

"Number <write the number> is inside the interval"

Solution

```
x = input()

message = "Number " + x + " is "

x = int(x)

if (x < 0 or x > 10):
    message += "NOT "

message += "inside the interval"

print(message)
```

Exercise

Write a program that reads the time of a clock and writes the time increased in a second.

The input and output must follow the following format: `05:23:52`
that is, hours, minutes, and seconds with two digits and separated by a colon.

First version:

- Read the time in the given format
- Display the read time in the given format

Second version:

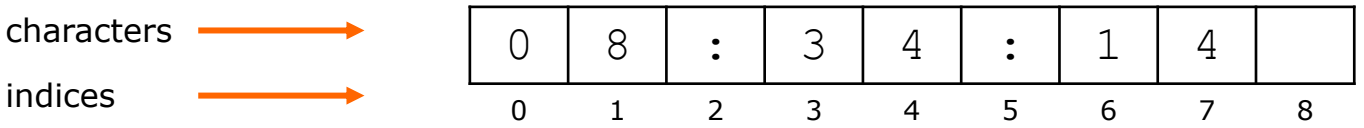
- Increase the time in 1 second, controlling that sometimes we also need to increase minutes and hours.

Remember what a string is

- A string is a sequence of characters.

```
In [1]: time = input("Enter a time in format hh:mm:ss = ")

Enter a time in format hh:mm:ss = 08:34:14
```

- How is it stored?



- Operator [ ] : access to a position

```
In [2]: time[2]
Out[2]: ':'
```

e.g. 3rd character (index 2)

- Operator [n:m] returns the part of the string from the n-th character (included) to the m-th (not included):

```
In [3]: time[0:2]
Out[3]: '08'
```

Solution (I)

```
# Input
time = input("Enter the time with format hh:mm:ss = ")


hh = int(time[0:2])
mm = int(time[3:5])
ss = int(time[6:8])

#Process
ss+=1

#Output
new_time = str(hh)+":"+str(mm)+":"+str(ss)

print(new_time)
```

Solution (II)

```
# Input
time = input("Enter the time with format hh:mm:ss = ")

hh = int(time[0:2])
mm = int(time[3:5])
ss = int(time[6:8])

#Process
ss+=1

#Output
new_time =""

# Format output when values < 10
if (hh<10):
    new_time += "0"
new_time += str(hh) + ":"
if (mm<10):
    new_time += "0"
new_time += str(mm) + ":"
if (ss<10):
    new_time += "0"
new_time += str(ss)

print(new_time)
```

Solution (III)

```
# Input
time = input("Enter the time with format hh:mm:ss = ")

hh = int(time[0:2])
mm = int(time[3:5])
ss = int(time[6:8])

#Process
ss+=1

if (ss == 60):
    ss = 0
    mm += 1
if (mm == 60):
    mm =0
    hh += 1
if (hh == 24):
    hh = 0

#Output
new_time =""

# Format output when values < 10
if (hh<10):
    new_time += "0"
new_time += str(hh) + ":"
if (mm<10):
    new_time += "0"
new_time += str(mm) + ":"
if (ss<10):
    new_time += "0"
new_time += str(ss)

print(new_time)
```

Exercise: Area and Perimeter of a circle

Write a program to compute the area and the perimeter of a circle:

- The program must ask the user to enter the radius of a circle and compute the area and the perimeter. After that, the program will show the result on screen.

- The program must check that the radius is a positive value and do the operations to compute the area and the perimeter only if the user has entered a positive radius.

Solution

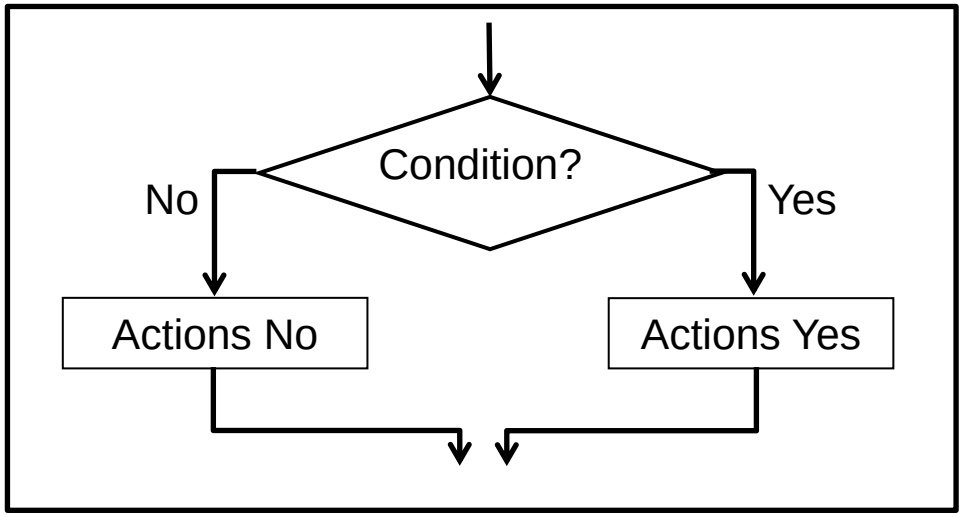
Example: Calculate the area and perimeter of a circle

```
# Program to compute area and perimeter of a circle

PI = 3.141592    # Definition of constant PI

#Input
radius = float(input("Enter the radius of the circle: "))

#Process
if (radius > 0):      # Check if radius is positive
    area = PI * radius ** 2
    perimeter = 2 * PI * radius
    #Output
    print("The area of the circle is: ", area)
    print("The perimeter of the circumference is: ", perimeter)
else:
    #Output
    print("ERROR: The radius must be a positive value.")
```

Conditional structures: double alternative

In the double alternative, we have a block of actions that is run if a certain condition is met, and another block of actions is run otherwise.

Flowchart



Syntax

```
if (<condition>):

  <actions YES>

else:

  <actions NO>
```

Exercise: Positive or negative?

Write a program that displays the message "`Positive`", if a positive number (including 0) has been entered from the keyboard, and displays the message "`Negative`", otherwise.

Solution

```
"""
Program that displays the message:
    ”Positive”  if a positive number (including 0) has
                 been read from keyboard
    ”Negative”  otherwise.
"""

#Input
num = int(input("Enter an integer: "))

#Process
if (num >= 0):
    #Output
    print("Positive")
else:
    #Output
    print("Negative")
```

Exercise: Vowel or consonant?

Write a program that asks the user to enter a letter and then tells if it is a vowel or a consonant.

Solution

```
"""
Program that displays the message:
    ”Vowel”     if user enters a,e,i,o,u
    ”Consonant” otherwise
"""

#Input
letter = input("Enter a character: ")

#Process
if (letter=="a" or
    letter=="e" or
    letter=="i" or
    letter=="o" or
    letter=="u"):
    #Output
    print("Vowel")
else:
    #Output
    print("Consonant")
```

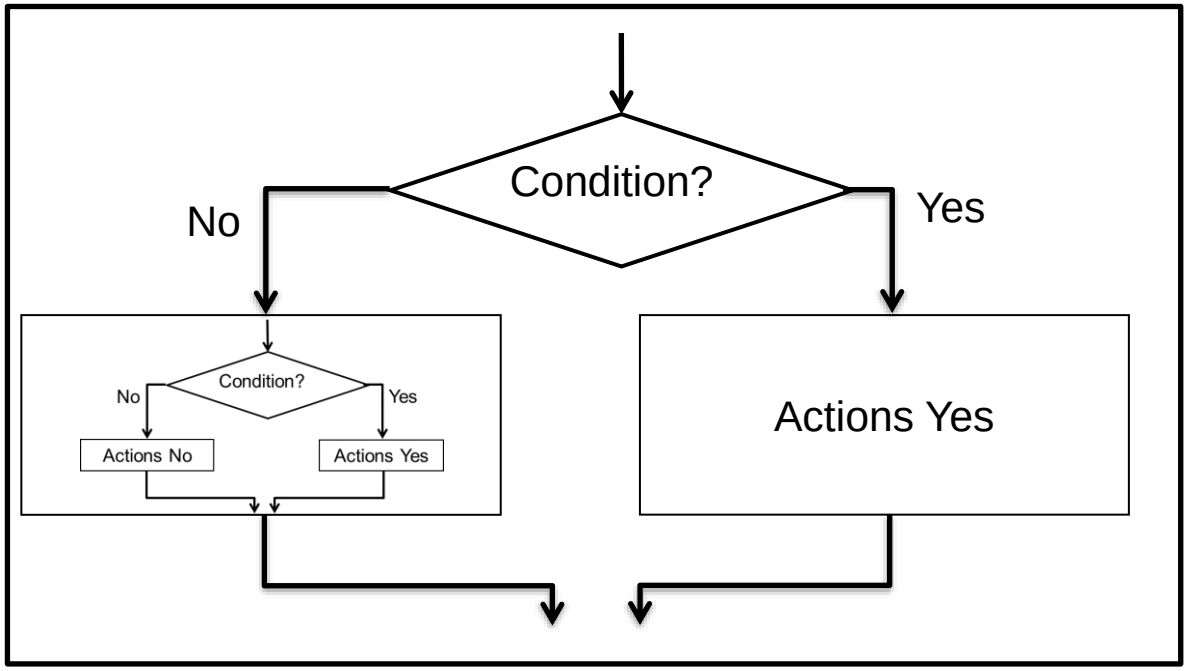
Solution

```
"""
Program that displays the message:
    ”Vowel”     if user enters a,e,i,o,u
    ”Consonant” otherwise
"""

#Input
letter = input("Enter a character: ")

#Process
if letter in "aeiouAEIOU":
    #Output
    print("Vowel")
else:
    #Output
    print("Consonant")
```

Double alternative particular case: nested if-else

The structures that we have explained are like blocks that can be used inside others, meanwhile we maintain their structure.

Flowchart



Syntax

```
if (<condition>):
  <actions YES>
else:
  if (<condition>):
    <actions YES>
  else:
    <actions NO>
```

IMPORTANT: Indentation!

Exercise: Positive, negative or zero

Write a program that asks for an integer and displays if the number is "`Positive`", "`Negative`" or "`Zero`".

Solution

```
"""
Program that displays the message:
    ”Positive” if the user enters a number > 0
    ”Negative” if the number is < 0
    "Zero"     otherwise
"""

#Input
num = int(input("Enter an integer: "))

#Process
if (num > 0):
    #Output
    print("Positive")
else:
    if (num < 0):
        #Output
        print("Negative")
    else:
        #Output
        print("Zero")
```

Exercise: Salary calculator

Write program that calculates the salary of a worker.

The program reads two values:

- base salary in euros (float), and
- years of service (int).

The program must calculate and show the final salary. This is calculated as the base salary plus a percentage of the base salary based on the years in the company of service.

The percentage of increase based on service is as follows:

- Less than 3 years: increase of 1%
- Between 3 and 5 years: increase of 2%
- More than 5 years: increase of 3.5%

Solution

```
"""
Program to compute the final salary depending on the service years
    Less than 3 years:      increase 1%
    Between 3 and 5 years:  increase 2%
    More than 5 years:      increase 3,5%
"""

#Input
base_salary = float(input("Enter the base salary: "))
service = int(input("Enter the years of service: "))


#Process
if (service < 3):
    final_salary = base_salary * 1.01
else:
    if (service < 5):
        final_salary = base_salary * 1.02
    else:
        final_salary = base_salary * 1.035

#Output
print("The final salary is: ", final_salary)
```

Exercise: Final grades

Write a program that converts a **numerical mark to a grade** (A with Honors, A, B, C, or Fail).

The program must ask the user to **enter a numerical mark** that should be a real number between 0 and 10.

After that, the program must **show the final grade** according to the following correspondence:

Fail	mark in the interval [0,5)
C	mark in the interval [5,7)
B	mark in the interval [7,9)
A	mark in the interval [9,10)
A with Honors	mark is 10

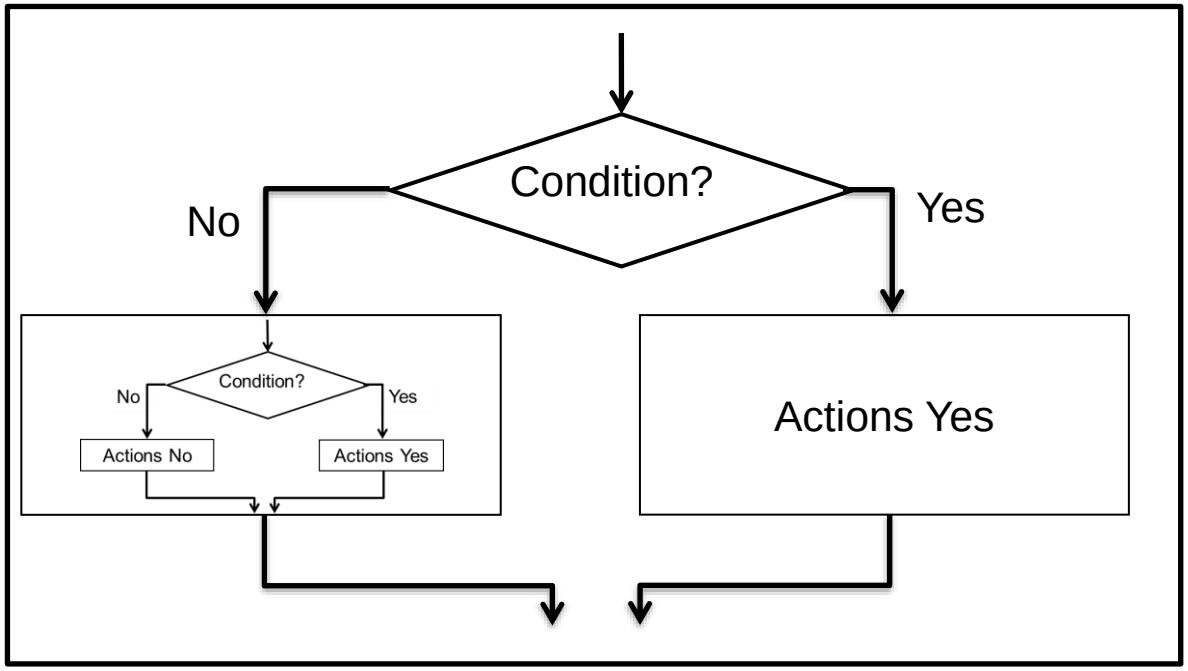
Solution

```
"""
Program that computers the final grade from the numerical mark
"""

#Input
mark = float(input("Enter the numerical mark: "))


#Process
if(0<=mark<=10):
    if (mark < 5):
        grade = "Fail"
    else:
        if (mark < 7):
            grade = "C"
        else:
            if (mark < 9):
                grade = "B"
            else:
                if (mark < 10):
                    grade = "A"
                else:
                    grade = "A with Honors"
    #Output
    print("The final grade is: ", grade)
else:
    print("Error, the final grade must be in [0,10]")
```

Conditional structures: nested alternative

In the case of nested if-else we can use the abbreviation elif.
Depending on the conditions, only one branch will be executed.
There is no limit to the amount of elif.

Flowchart



Syntax

```
if (<condition>):
  <actions YES>
elif (<condition>):
    <actions YES>
else:
    <actions NO>
```

Exercise: Tasks Menu

Write a program that allows us to simulate a menu of three options:

```
Menu

1 - Task 1
2 - Task 2
3 - Task 3

Select a task:
```

Once the user has entered one of the options, the program will display a message indicating that the corresponding task is being done. These messages will have format: "Doing Task X", where X will be an integer corresponding to the selected option (1, 2 or 3).

If the user selects a task different to 1, 2 and 3, the program will display the following error message: "ERROR: Wrong choice".

Solution

```
"""
This code implements a menu
"""

print(“1 - Task 1\n2 - Task 2\n3 - Task 3")
selection = int(input("Select a task: "))

if (selection == 1):
    print("Doing Task 1")
elif (selection == 2):
    print("Doing Task 2")
elif (selection == 3):
    print("Doing Task 3")
else:
    print("ERROR: Wrong choice")
```

Exercise

Write a program that asks for a letter

First, the program will have to check that it is a letter ["a" ... "z"] or ["A" ... "Z"].

If it is a letter, the program will tell if it is a vowel or a consonant in uppercase or lowercase.

In any other case, the program will show a message warning that the character is not allowed.

Output:

- "Lowercase vowel" if you enter a,e,i,o,u
- "Uppercase vowel" if you enter A,E,I,O,U
- "Lowercase consonant" if you enter b,c,d,f, ... x,y,z
- "Uppercase consonant" if you enter B,C,D,F, ... X,Y,Z
- "Character not allowed" in any other case

The ASCII Code (reminder)

The ASCII code is a Latin-based character code. It was created in 1963 by the American Committee on Standards (source: Wikipedia).

Essentially, it establishes a correspondence between the 256 integers that fit in a byte (8 bits of information = 2^8 possible combinations) and the alphabet.

The most interesting values for us are the letters "a" and "A", and the digit "0". These characters have codes 97, 65 and 48 respectively.

The interest in choosing these comes from the fact that the entire lowercase alphabet comes after letter "a", so "b" has code 98, "c" has code 99, and so on until "z", which has code 122.

The same happens for capital letters: "B" is 66, "C" on 67, and so on until "Z", which is 90.

Numbers from "0" to "9" have codes from 48 to 57.

The ASCII Code (reminder)

Codi	Car.	Codi	Car.	Codi	Car.	Codi	Car.	Codi	Car.	Codi	Car.	Codi	Car.
32		64	@	96	`	128	€	160		192	À	224	à
33	!	65	A	97	a	129	□	161	¡	193	Á	225	á
34	"	66	B	98	b	130	‚	162	¢	194	Â	226	â
35	#	67	C	99	c	131	ƒ	163	£	195	Ã	227	ã
36	$	68	D	100	d	132	„	164	¤	196	Ä	228	ä
37	%	69	E	101	e	133	…	165	¥	197	Å	229	å
38	&	70	F	102	f	134	†	166	¦	198	Æ	230	æ
39	'	71	G	103	g	135	‡	167	§	199	Ç	231	ç
40	(	72	H	104	h	136	ˆ	168	¨	200	È	232	è
41	)	73	I	105	i	137	‰	169	©	201	É	233	é
42	*	74	J	106	j	138	Š	170	ª	202	Ê	234	ê
43	+	75	K	107	k	139	‹	171	«	203	Ë	235	ë
44	,	76	L	108	l	140	Œ	172	¬	204	Ì	236	ì
45	-	77	M	109	m	141	□	173		205	Í	237	í
46	.	78	N	110	n	142	Ž	174	®	206	Î	238	î
47	/	79	O	111	o	143	□	175	¯	207	Ï	239	ï
48	0	80	P	112	p	144	□	176	°	208	Ð	240	ð
49	1	81	Q	113	q	145	‘	177	±	209	Ñ	241	ñ
50	2	82	R	114	r	146	’	178	²	210	Ò	242	ò
51	3	83	S	115	s	147	"	179	³	211	Ó	243	ó
52	4	84	T	116	t	148	"	180	´	212	Ô	244	ô
53	5	85	U	117	u	149	•	181	µ	213	Õ	245	õ
54	6	86	V	118	v	150	–	182	¶	214	Ö	246	ö
55	7	87	W	119	w	151	—	183	·	215	×	247	÷
56	8	88	X	120	x	152	˜	184	¸	216	Ø	248	ø
57	9	89	Y	121	y	153	Ö	185	¹	217	Ù	249	ù
58	:	90	Z	122	z	154	š	186	º	218	Ú	250	ú
59	;	91	[	123	{	155	›	187	»	219	Û	251	û
60	<	92	\	124	|	156	œ	188	¼	220	Ü	252	ü
61	=	93	]	125	}	157	□	189	½	221	Ý	253	ý
62	>	94	^	126	~	158	ž	190	¾	222	Þ	254	þ
63	?	95	_	127	□	159	Ÿ	191	¿	223	ß	255	ÿ

Each character corresponds to a number between 0 and 255.

On the keyboard:
ALT + Num → character from table

Example: ALT + 169 → ©

Python

```
print(chr(65))   → 'A'
print(ord("A")) → 65
```

Conditional structures: multiple alternative

```
"""
Program that displays the message:
    "Lowercase vowel"  if you enter a,e,i,o,u
    "Uppercase vowel"  if you enter A,E,I,O,U
    "Lowercase consonant"  if you enter b,c,d,f, ... x,y,z
    "Uppercase consonant"  if you enter B,C,D,F, ... X,Y,Z
    "Character not allowed"  in any other case
"""

#Input
letter = input("Enter a character: ")
numerical_code = (ord(letter))

#Proces
pre  = ""
post = ""
if ((numerical_code >= ord("A") and numerical_code <= ord("Z"))):
    pre = "Uppercase"
    if letter in "AEIOU":
        post = "vowel"
    else:
        post = "consonant"
elif ((numerical_code >= ord("a") and numerical_code <= ord("z"))):
    pre = "Lowercase"
    if letter in "aeiou":
        post = "vowel"
    else:
        post = "consonant"
else:
    pre = "Character not allowed"

#Output
print(pre,post)
```

Questions?