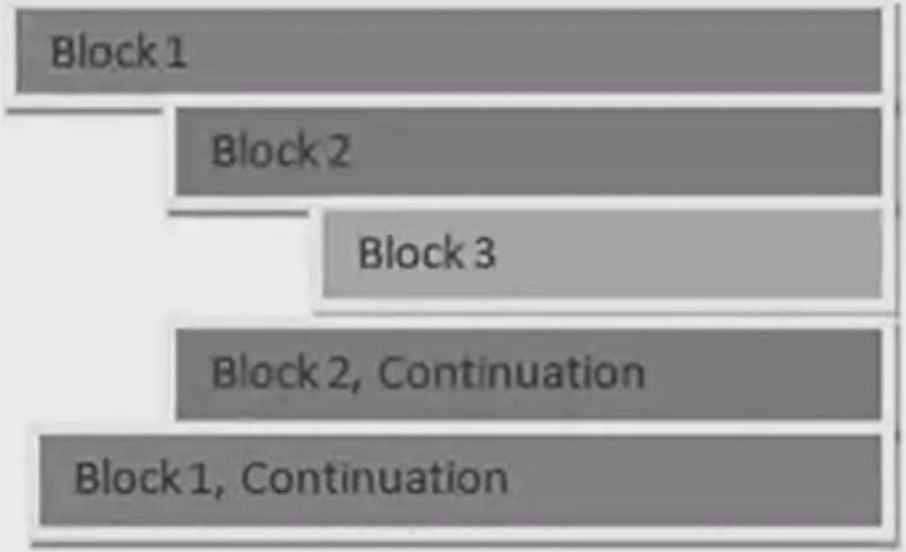
REC

Indentation



Code blocks and indentation in Python.

You

REC

Comments

- Comments are an important part of any program. A comment is a text that describes what the program or a particular part of the program is trying to do and is ignored by the Python interpreter.
- Comments are used to help you and other programmers understand, maintain, and debug the program.
- Python uses two types of comments: single-line comment

```
#This is single line Python comment
```

- multiline comments.

```
#This is
#multiline comments
#in Python
```

You

REC

Reading Input

- In Python, *input()* function is used to gather data from the user. The syntax for input function is,

variable_name = input([prompt])

- >>> person = input("What is your name?")
- What is your name? Carrey
- >>> person
- 'Carrey'

You

REC

Print Output

- The *print()* function allows a program to display text onto the console.
- print("Hello World!!")

  Hello World!!

- Two major string formats which are used inside the *print()* function to display the contents onto the console
- 1. str.format()
- 2. f-strings

format() Method

- The syntax for *format()* method is, *str.format(p0, p1, ..., k0=v0, k1=v1, ...)*

p0, p1,... are called as positional arguments and, k0, k1,... are keyword arguments with their assigned values of v0, v1,... respectively.

- Formatted strings or f-strings were introduced in Python 3.6. A *f-string* is a string literal that is prefixed with "f". These strings may contain replacement fields, which are expressions enclosed within curly braces {}. The expressions are replaced with their values.

Example:

```
USN = input("Enter your USN: ")
Name = input("Enter your Name: ")
print("Student USN is {0} and  name {1}".format(USN, Name))
print("Student name is {1} and USN is {0}".format(USN, Name))
print("Student belongs to {Section}, ECE".format(Section = "6ABCD"))
print(f"Student USN {USN} and {Name}")
```

Module 1-Programming in Python - PowerPoint

DRAWING TOOLS

FILE HOME INSERT DESIGN TRANSITIONS ANIMATIONS SLIDE SHOW REVIEW VIEW FORMAT

REC

Clipboard Slides Font Paragraph Drawing

Type Conversions

You can explicitly cast, or convert, a variable from one type to another.

The *int()* Function
The *float()* Function
The *str()* Function
The *chr()* Function
The *complex()* Function
The *ord()* Function
The *hex()* Function
The *oct()* Function

The *type()* Function and is Operator
type(object)
The *type()* function returns the data type of the given object

```
type(6.4)
<class 'float'>
```

11-09-2020

SLIDE 39 OF 80 ENGLISH (INDIA) NOTES COMMENTS

Python 3.8.1 Shell

File Edit Shell Debug Options Window Help

```
Python 3.8.1 (tags/v3.8.1:1b293b6, Dec 18 2019, 22:39
:24) [MSC v.1916 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license()" fo
r more information.
>>> inpu("usn")
Traceback (most recent call last):
  File "<pyshell#0>", line 1, in <module>
    inpu("usn")
NameError: name 'inpu' is not defined
>>> input("usn")
usn1ds12345
'1ds12345'
>>>
```

You

11-09-2020

Module 1-Programming in Python - PowerPoint

DRAWING TOOLS

FILE HOME INSERT DESIGN TRANSITIONS ANIMATIONS SLIDE SHOW REVIEW VIEW FORMAT

REC

Clipboard Slides Font Paragraph Drawing

Type Conversions

You can explicitly cast, or convert, a variable from one type to another.

The *int()* Function
The *float()* Function
The *str()* Function
The *chr()* Function
The *complex()* Function
The *ord()* Function
The *hex()* Function
The *oct()* Function

The *type()* Function and is Operator
type(object)
The *type()* function returns the data type of the given object
type(6.4)
<class 'float'>

11-09-2020

SLIDE 39 OF 80 ENGLISH (INDIA) NOTES COMMENTS

Python 3.8.1 Shell

File Edit Shell Debug Options Window Help

```
Python 3.8.1 (tags/v3.8.1:1b293b6, Dec 18 2019, 22:39
:24) [MSC v.1916 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license()" fo
r more information.
>>> inpu("usn")
Traceback (most recent call last):
  File "<pyshell#0>", line 1, in <module>
    inpu("usn")
NameError: name 'inpu' is not defined
>>> input("usn")
usn1ds12345
'1ds12345'
>>> input("num")
num5
'5'
>>> int(input("num"))
num5
5
>>> int(12.34)
12
>>> int(True)
1
>>> int(10+20j)
Traceback (most recent call last):
  File "<pyshell#6>", line 1, in <m
    int(10+20j)
TypeError: can't convert complex to
>>>
```

You

11-09-2020

Dec	Hx	Oct	Char		Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr
0	0	000	NUL	(null)	32	20	040		Space	64	40	100	@	@	96	60	140	`	`
1	1	001	SOH	(start of heading)	33	21	041	!	!	65	41	101	A	A	97	61	141	a	a
2	2	002	STX	(start of text)	34	22	042	"	"	66	42	102	B	B	98	62	142	b	b
3	3	003	ETX	(end of text)	35	23	043	#	#	67	43	103	C	C	99	63	143	c	c
4	4	004	EOT	(end of transmission)	36	24	044	$	$	68	44	104	D	D	100	64	144	d	d
5	5	005	ENQ	(enquiry)	37	25	045	%	%	69	45	105	E	E	101	65	145	e	e
6	6	006	ACK	(acknowledge)	38	26	046	&	&	70	46	106	F	F	102	66	146	f	f
7	7	007	BEL	(bell)	39	27	047	'	'	71	47	107	G	G	103	67	147	g	g
8	8	010	BS	(backspace)	40	28	050	(	(	72	48	110	H	H	104	68	150	h	h
9	9	011	TAB	(horizontal tab)	41	29	051	)	)	73	49	111	I	I	105	69	151	i	i
10	A	012	LF	(NL line feed, new line)	42	2A	052	*	*	74	4A	112	J	J	106	6A	152	j	j
11	B	013	VT	(vertical tab)	43	2B	053	+	+	75	4B	113	K	K	107	6B	153	k	k
12	C	014	FF	(NP form feed, new page)	44	2C	054	,	,	76	4C	114	L	L	108	6C	154	l	l
13	D	015	CR	(carriage return)	45	2D	055	-	-	77	4D	115	M	M	109	6D	155	m	m
14	E	016	SO	(shift out)	46	2E	056	.	.	78	4E	116	N	N	110	6E	156	n	n
15	F	017	SI	(shift in)	47	2F	057	/	/	79	4F	117	O	O	111	6F	157	o	o
16	10	020	DLE	(data link escape)	48	30	060	0	0	80	50	120	P	P	112	70	160	p	p
17	11	021	DC1	(device control 1)	49	31	061	1	1	81	51	121	Q	Q	113	71	161	q	q
18	12	022	DC2	(device control 2)	50	32	062	2	2	82	52	122	R	R	114	72	162	r	r
19	13	023	DC3	(device control 3)	51	33	063	3	3	83	53	123	S	S	115	73	163	s	s
20	14	024	DC4	(device control 4)	52	34	064	4	4	84	54	124	T	T	116	74	164	t	t
21	15	025	NAK	(negative acknowledge)	53	35	065	5	5	85	55	125	U	U	117	75	165	u	u
22	16	026	SYN	(synchronous idle)	54	36	066	6	6	86	56	126	V	V	118	76	166	v	v
23	17	027	ETB	(end of trans. block)	55	37	067	7	7	87	57	127	W	W	119	77	167	w	w
24	18	030	CAN	(cancel)	56	38	070	8	8	88	58	130	X	X	120	78	170	x	x
25	19	031	EM	(end of medium)	57	39	071	9	9	89	59	131	Y	Y	121	79	171	y	y
26	1A	032	SUB	(substitute)	58	3A	072	:	:	90	5A	132	Z	Z	122	7A	172	z	z
27	1B	033	ESC	(escape)	59	3B	073	;	;	91	5B	133	[	[	123	7B	173	{	{
28	1C	034	FS	(file separator)	60	3C	074	<	<	92	5C	134	\	\	124	7C	174			|
29	1D	035	GS	(group separator)	61	3D	075	=	=	93	5D	135	]	]	125	7D	175	}	}
30	1E	036	RS	(record separator)	62	3E	076	>	>	94	5E	136	^	^	126	7E	176	~	~
31	1F	037	US	(unit separator)	63	3F	077	?	?	95	5F	137	_	_	127	7F	177	&#127;	DEL

Source: www.LookupTables.com

Would you like to make Internet Explorer your default browser? Yes No

REC

REC D You pres

You

REC

is , is not

- The operators *is* and *is not* are identity operators. Operator *is* evaluates to *True* if the values of operands on either side of the operator point to the same object and *False* otherwise.
- The operator *is not* evaluates to *False* if the values of operands on either side of the operator point to the same object and *True* otherwise.
- >>> a=2
- >>> b=2
- >>> a is b

  True
- >>> a is not b

  False
- >>> c=4
- >>> a is not c

  True

You

REC

Control Flow Statements

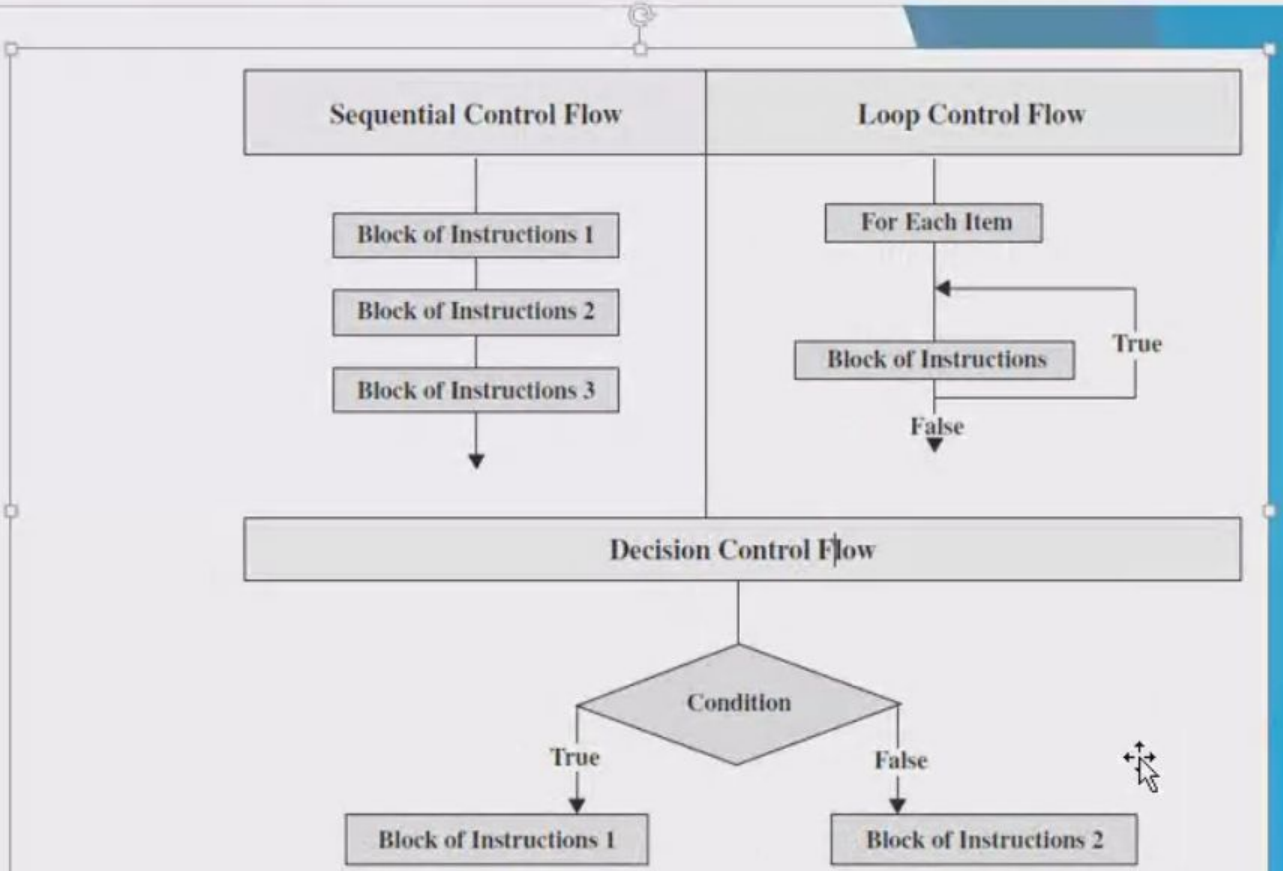


FIGURE 3.1
Forms of control flow statements.

REC

control flow statements

The control flow statements in Python Programming Language are

- 1. **Sequential Control Flow Statements:** This refers to the line by line execution, in which the statements are executed sequentially, in the same order in which they appear in the program.

- 2. **Decision Control Flow Statements:** Depending on whether a condition is True or False, the decision structure may skip the execution of an entire block of statements or even execute one block of statements instead of other (if, if...else and if...elif...else).

- 3. **Loop Control Flow Statements:** This is a control structure that allows the execution of a block of statements multiple times until a loop termination condition is met (*for* loop and *while* loop). Loop Control Flow Statements are also called Repetition statements or Iteration statements.

You