If. Then. Else.

Introduction to Computer Programming

Dr. Paul Vrbik

September 25, 2018

In-line

We have already seen a simple if-statement.

```
>>> x = 7 if "a" < "b" else 6
>>> x
7
>>> y = 0 if x < 7 else x
>>> y
7
>>> def f(x):
...   return "hello" if x > 0 else "world"
>>> f(1)
"hello"
```

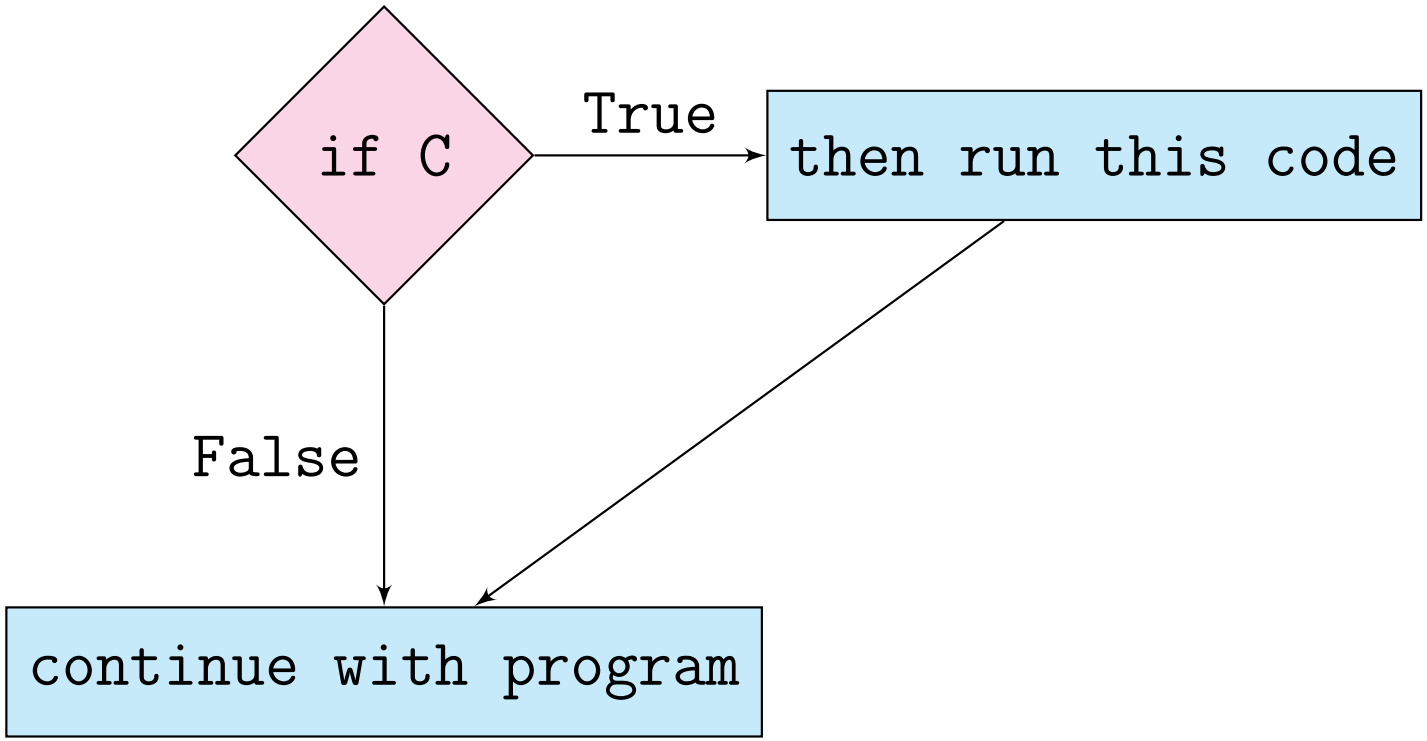
Definition (If-statement)

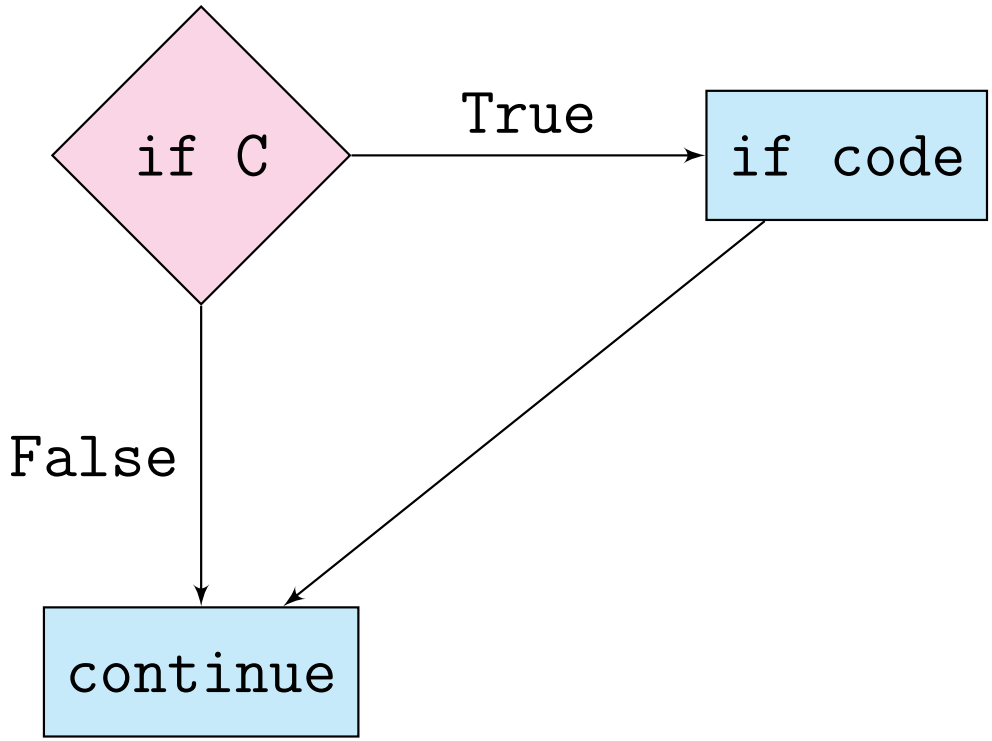
Given a condition or predicate statement C (i.e. something that evaluates to a boolean) an if-statement is a control structure that executes a block of code when C is True.

If-Then

```
if <cond>:
    <code executed when cond == True>
      ⋮
```

In Python spaces matter — only code indented within an if-statement gets executed.


if C
True
then run this code
False
continue with program


if C
True
if code
False
continue

```
>>> x = 1
>>> if 0 == 7:
...     x = x + 1
>>> x
1


>>> x = 1/1
>>> if type(x) is int:
...     x = x +1
>>> x
1.0
```

```
>>> xs = "hello world"
>>> if ‘h’ in xs:
...     xs = "goodbye" + xs[-6:]
>>> xs
‘goodbye world’

>>> xs = "hello world"
>>> if ‘a’ in xs:
...     xs = ""
>>> xs
‘hello world’
```

```
>>> def foo(x):
...     if x > 0:
...         print("Positive")
...     if x > 10**5:
...         print("Large positive")

>>> ans = foo(10**6)
Positive
Large postive

>>> type(ans)
<class 'NoneType'>
```

```
>>> def bar(x):
...     if x > 0:
...         return "Positive"
...     if x > 10**5:
...         return "Large positive"

>>> ans = bar(10**6)
>>> type(ans)
<class 'str'>
>>> ans
'Positive'
```

```
>>> if False:
...     ans = 0
>>> ans
NameError: name 'ans' is not defined
```

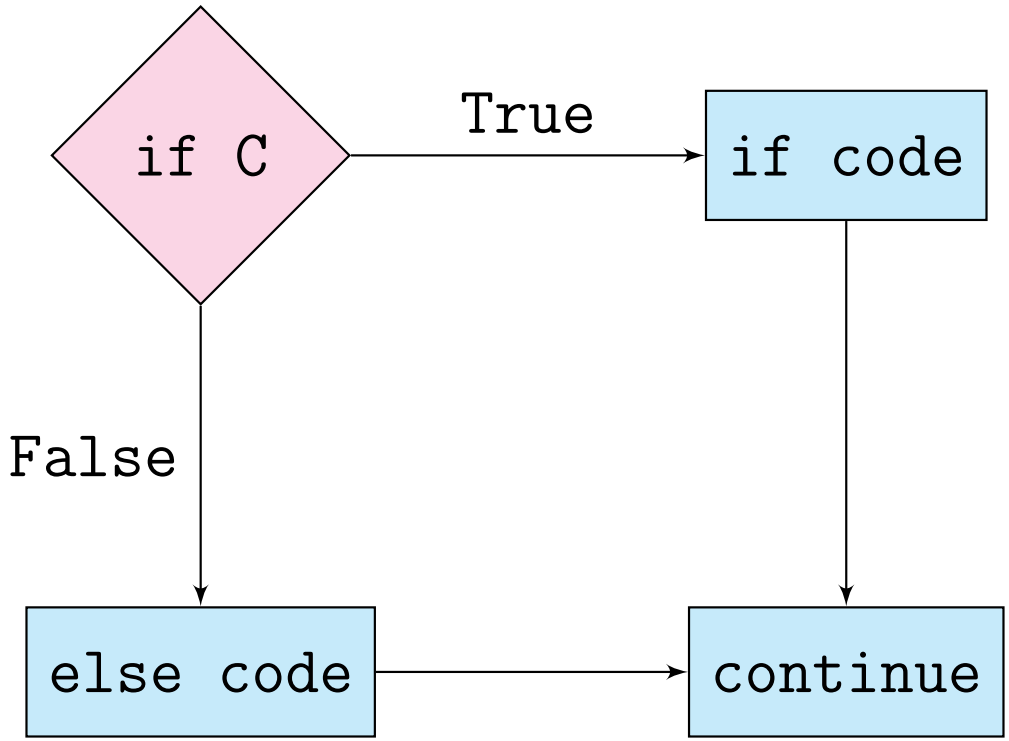
The code in the if-statement is skipped and therefore `ans` does not get set.

```
>>> if balance >= 0:
...     in_the_black = True
...     in_the_red = False

>>> if balance < 0:
...     in_the_black = False
...     int_the_red = True
```

This makes our code longer and checks the condition twice.

If-Then-Else

```
if <cond>:
    <code>
else:
    <code>
```


if C
True
if code
False
else code
continue

```
>>> if balance >= 0:
...     in_the_black = True
...     in_the_red = False
... else:
...     in_the_black = False
...     int_the_red = True
```

Now “extra” code executes regardless of the truth of C.

Question

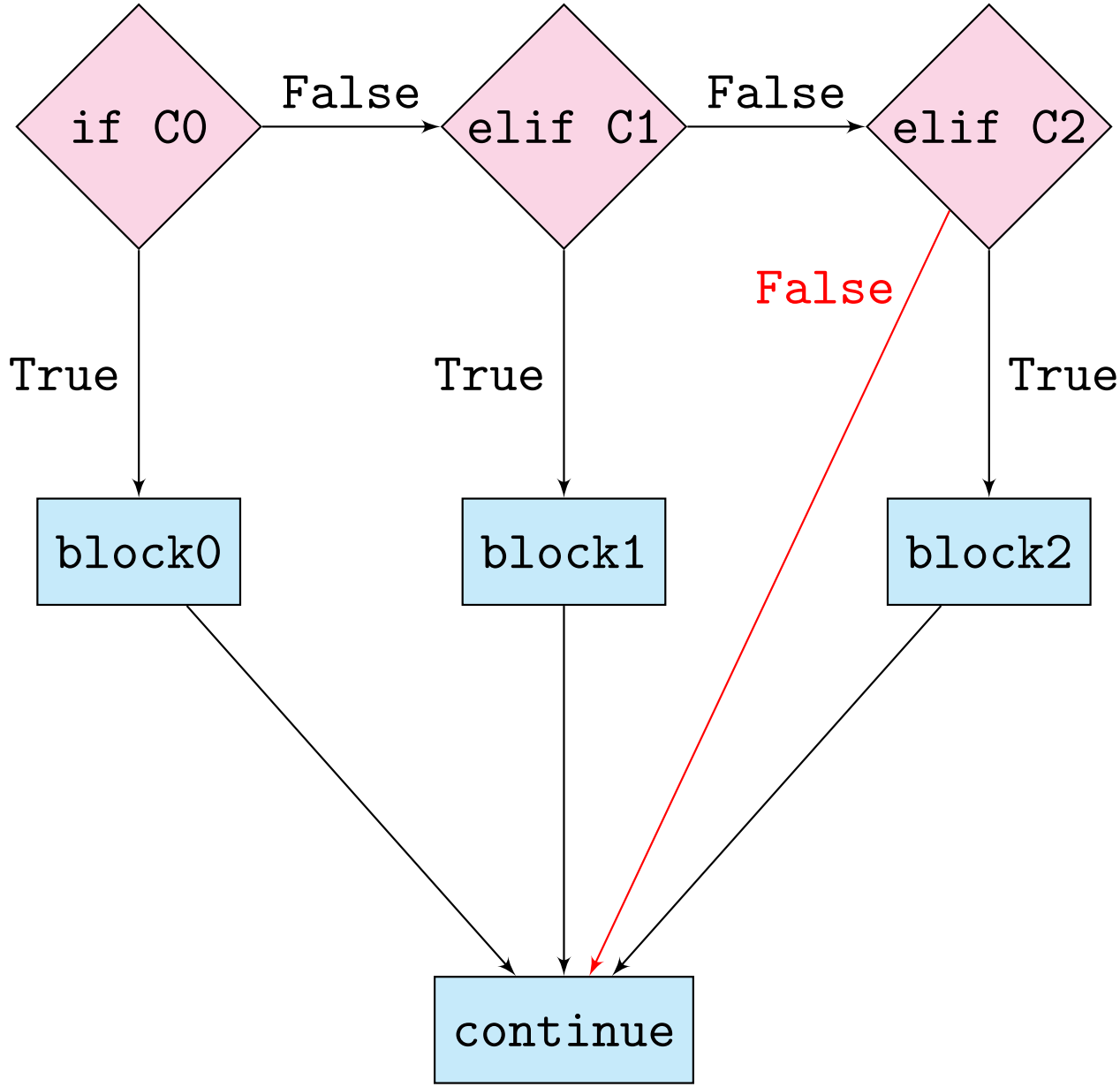
Write a function that returns the longest of two strings.

```
def longer_string(string1:str, string2:str) -> str:
    """
    Returns the longer of two strings and string1
    when tied for length.
    >>> longer("fancy", "pants")
    'fancy'
    >>> longer(" ", "amazing")
    'amazing'
    """
    if len(string1)>=len(string2):
        return string1
    else:
        return string2
```

```
>>> if credit == "poor":
...     print("no credit for you")

>>> if credit == "good":
...     print("maybe credit for you")

>>> if credit == "excellent":
...     print("let me give you credit")

>>> if credit != "poor" and credit != "good" \
and credit != "excellent":
...     print("obtain credit score")
```

Else-If

```
if <cond0>:
    <code>
elif <cond1>:
    <code>
  :
elif <condN>:
    <code>
```

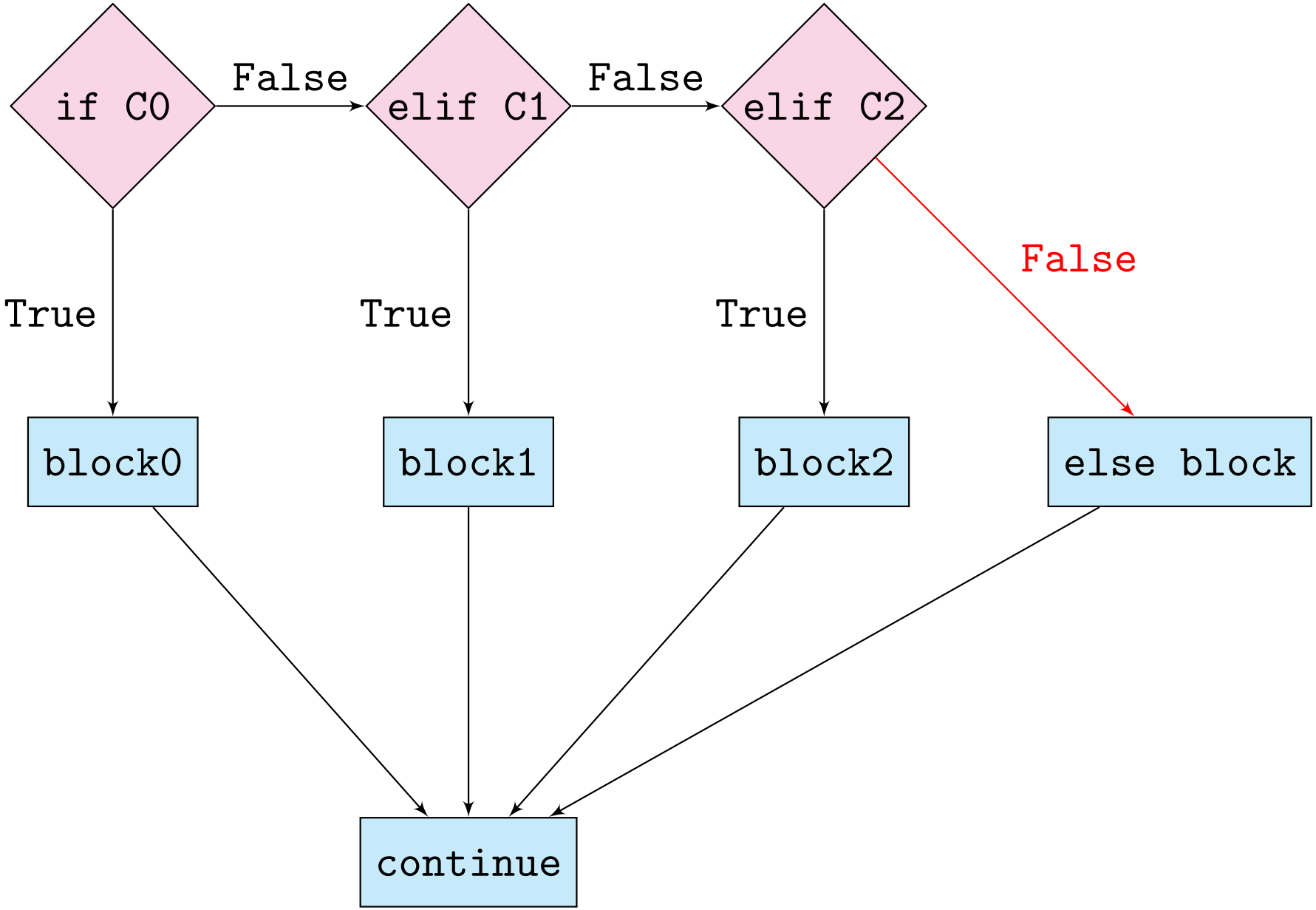

if C0
False
elif C1
False
elif C2
True
True
False
True
block0
block1
block2
continue

```
>>> if credit == "poor":
...     print("no credit for you")
... elif credit == "good":
...     print("maybe credit for you")
... elif credit == "excellent":
...     print("let me give you credit")
... else:
...     print("obtain credit score")
```

```
>>> x = True
>>> y = False

>>> if not x:
...     ans = "panda"
>>> elif x and y:
...     ans = "snake"
>>> elif not x or y:
...     ans = "badger"

>>> ans
NameError: name ‘ans’ is not defined
```

Else-If-Else

```
if <cond0>:
    <code>
elif <cond1>:
    <code>
  ⋮
elif <condN>:
    <code>
else:
    <code>
```


if C0
False
elif C1
False
elif C2
True
True
True
False
block0
block1
block2
else block
continue

```
>>> (x, y) = (True, False)


>>> if not x:
...     ans = "panda"
>>> elif x and y:
...     ans = "snake"
>>> elif not x or y:
...     ans = "badger"
>>> else:
...     ans = "man_bear_pig"
>>> ans
‘man_bear_pig’
```

Nesting If-Statements

```
>>> x = float('inf')
>>> if x > 0:
...     print("x is positive")
...     if x > 1:
...         print("x is not small")
...         if x > 10**10:
...             print("x is pretty big")
```

Next Time

1. Simplifying conditions – ‘code refactoring.’