```
#
# Find the remaining string, (Please see the information given below the skeleton program)
#

# Hint :
#    try the following in interpreter
#        >>> str = ''
#        >>> str = str + 'a',
#        >>> str
# (Basically str1+str2 returns the result of concatenating str1 and str2)

def string_after_firstOccurrence(str,ch):



s = raw_input("Enter the string")
c = raw_input("Enter the character")
str1=string_after_firstOccurrence(s,c)
print "The remaining string after the first occurrence of ",c," is ",str1

#    Example sets
#
#    1) str = file.txt.pdf
#       ch = .
#       Observable_Output : The remaining string after the first occurrence of . is txt.pdf
#

#    2) str = aardvark.txt
#       ch = a
#       Observable_Output : The remaining string after the first occurrence of a is ardvark
#

#    3) str = polynomial-function
#       ch = o
#       Observable_Output : The remaining string after the first occurrence of o is
lynomial-function
#


#     Trace format
#
#    Example set 1
#
#     Step    program_line    What_happens_inside_the_computer
#        1        9              s = "file.txt.pdf"
#        2        10             c = '.'
#        3        11             calls string_after_firstOccurrence(s,c) ==>
remaining_string('file.txt.pdf','.')
#
#     Step    program_line    Observable_Output
#        1
#


# Note :-
#    In all programs in this set, the following rules hold:
#    1) You can only add new code and not delete any line/character
#    2) You have to trace the code by hand on the example sets given below the program
#    3) The final trace must be available on the example inputs below the program
#
#    Besides the above rules, the spirit/manner in which you must develop the code is as
follows,
#        First you will have an idea then you code it up and then you run/trace the code on
the example
```

```
#             sets and then you will realise the mistake(s) made. Then either you realise
that the initial idea itself
#             was wrong and you change tracks, or, you refine the code and eliminate the
bugs.
#
#        Finally, I want to see the efforts taken by you in the trace below the final
program.
#
```

```
#
# Find the index of the first occurrence of a character
#

# Returns the index of the first occurrence of ch in str
#    If ch does not occur in str then return -1 (an invalid index, if seen from left to
right)

def first_occurrence(str,ch):


s = raw_input("Enter the string")
c = raw_input("Enter the character")
i = first_occurrence(s,c)

if(i==-1):
    print 'the character ',c,' is not in the string \"',s.'\"'
else:
    print 'The first occurrence of ',s[i],' is at position ',i

#    Example sets
#
#    1) str = file.txt.pdf
#       ch = .
#       Observable_Output : The first occurrence of . is at position 4
#

#    2) str = aardvark.txt
#       ch = a
#       Observable_Output : The first occurrence of a is at position 0
#

#    3) str = polynomial
#       ch = .
#       Observable_Output : The character . is not in the string "polynomial"
#


#     Trace format
#
#    Example set 1
#
#     Step    program_line    What_happens_inside_the_computer
#        1        9              s = "file.txt.pdf"
#        2        10             c = '.'
#        3        11             first_occurrence(s,c) ==> first_occurrence
('file.txt.pdf','.')
#
#     Step    program_line    Observable_Output
#        1
#
```

```
#
# Find remaining string using first_occurrence(str,ch)
#

# copy the code you wrote in 2.py here
def first_occurrence(str,ch):


# Use the first_occurrence(str,ch) to define string_after_firstOccurrence(str,ch)
#    If the character ch does not occur in str, then we return '' (the empty string)
#      else as in 1.py, we return the remaining string after ch.
def string_after_firstOccurrence(str,ch):
    i = first_occurrence(str,ch)
    if(i==-1):
        return ''



s = raw_input("Enter the string")
c = raw_input("Enter the character")
str1=string_after_firstOccurrence(s,c)
print"The remaining string after the first occurrence of ",c," is ",str1


#    Example sets
#
#    1) str = file.txt.pdf
#       ch = .
#       Observable_Output : The remaining string after the first occurrence of . is txt.pdf
#

#    2) str = aardvark.txt
#       ch = a
#       Observable_Output : The remaining string after the first occurrence of a is ardvark
#
#    3) str = polynomial-function
#       ch = o
#       Observable_Output : The remaining string after the first occurrence of o is
lynomial-function
#


#     Trace format
#
#    Example set 1
#
#     Step    program_line    What_happens_inside_the_computer
#        1        9              s = "file.txt.pdf"
#        2        10             c = '.'
#        3        11             calls string_after_firstOccurrence(s,c) ==>
string_after_firstOccurrence('file.txt.pdf','.')
#

#     Step    program_line    Observable_Output
#        1
#
```

```
#
# Find the reverse of a given string
#

# Returns the reverse of a given string
def reverse(str):


s = raw_input("Enter the string")
rev=reverse(s)
print 'The reverse of the string \"",s," is \"',rev,'\"'


#    Example sets
#
#    1) str = pragmatic
#       Observable_Output : The reverse of the string "pragmatic" is "citamgarp"
#

#    2) str = citation
#       Observable_Output : The reverse of the string "citation" is "noitatic"


#     Trace format
#
#    Example set 1
#
#     Step    program_line    What_happens_inside_the_computer
#        1        9              s = "file.txt.pdf"
#        2        10             c = '.'
#        3        11             calls remaining_string(s,c) ==> remaining_string
('file.txt.pdf','.')
#
#     Step    program_line    Observable_Output
#        1
#
```

```
# Find the last occurrence of a particular character

# Returns the index of the last occurrence of ch in str
#    If ch does not occur in str then return -1 (an invalid index, if seen from left to
right)
def last_occurrence(str,ch):


s = raw_input("Enter the string")
c = raw_input("Enter the character")
i = last_occurrence(s,c)

if(i==-1):
    print 'the character ',c,' is not in the string \"',s.'\"'
else:
    print 'The last occurrence of ',s[i],' is at position ',i


#    Example sets
#
#    1) str = file.txt.pdf
#       ch = .
#       Observable_Output : The last occurrence of . is at position 4
#

#    2) str = aardvark.txt
#       ch = a
#       Observable_Output : The last occurrence of a is at position 5
#
#    3) str = polynomial
#       ch = .
#       Observable_Output : The character . is not in the string "polynomial"
#


#     Trace format
#
#    Example set 1
#
#     Step    program_line    What_happens_inside_the_computer
#        1        9             s = "file.txt.pdf"
#        2        10            c = '.'
#        3        11            last_occurrence(s,c) ==> last_occurrence
('file.txt.pdf','.')
#
#     Step    program_line    Observable_Output
#        1
#
```

```
# Using last occurrence and other functions created in this set,
#        return the remaining string after the last occurrence of the character.

# copy the code you wrote in 5.py here
def last_occurrence(str,ch):


# Use the first_occurrence(str,ch) to define string_after_firstOccurrence(str,ch)
#    If the character ch does not occur in str, then we return '' (the empty string)
#      else as in 1.py, we return the remaining string after ch.
def string_after_lastOccurrence(str,ch):
    i = last_occurrence(str,ch)
    if(i==-1):
        return ''



s = raw_input("Enter the string")
c = raw_input("Enter the character")
str1=string_after_lastOccurrence(s,c)
print"The remaining string after the last occurrence of ",c," is \"",str1,"\""

#    Example sets
#
#    1) str = file.txt.pdf
#       ch = .
#       Observable_Output : The remaining string after the last occurrence of . is "pdf"
#

#    2) str = aardvark.txt
#       ch = a
#       Observable_Output : The remaining string after the first occurrence of a is "rk.txt"
#
#    3) str = polynomial-function
#       ch = n
#       Observable_Output : The remaining string after the last occurrence of n is ""
#


#     Trace format
#
#    Example set 1
#
#     Step    program_line    What_happens_inside_the_computer
#        1        9              s = "file.txt.pdf"
#        2        10             c = '.'
#        3        11             calls string_after_firstOccurrence(s,c) ==>
string_after_firstOccurrence('file.txt.pdf','.')x
#
#     Step    program_line    Observable_Output
#        1
#
```

```
# Given a list, find if the list contains a given value

# Hint : Modify the functions defined in this set

# find(l,val) takes in a list of integers
#     and returns the first index of val in l if it exists
#                 and returns -1 otherwise
def find(l,val):



list=raw_input("enter a list")
list=eval(l)
x = raw_input("enter the value to search for")
i = find(l,x)

if(i==-1):
    print val," is not present in the list ",list
else:
    print list[i]," is present in the list ",list," at position ",i

#    Example sets
#
#    1) list = [10,20,5,10,25,5,24]
#       x = 5
#       Observable_Output : 5 is present in the list [10,20,5,10,25,5,24] at position 2
#

#    2) list = [5,10,25,24,29,30]
#       x = 30
#       Observable_Output : 30 is present in the list [10,20,5,10,25,5,24] at position 5
#

#    3) list = [5,10,25,24,12,31,53,63,98,75]
#       x = 4
#       Observable_Output : 4 is not present in the list [10,20,5,10,25,5,24]
#


#     Trace format
#
#    Example set 1
#
#     Step    program_line    What_happens_inside_the_computer
#        1        9              list = [10,20,5,10,25,5,24]
#        2        10             x=5
#        3        11             calls find(list,x) ==> find([10,20,5,10,25,5,24],5)
#
#     Step    program_line    Observable_Output
#        1
#
```