```
# File: MyStringFunctions.py
# Student: Jennifer Truong
# UT EID: Jat5244
# Course Name: CS303E
#
# Date Created: 3/31/2021
# Date Last Modified: 4/2/2021
# Description of Program: Building a string library and defining a collection of
functions on strings

def myAppend( str, ch ):
    # Return a new string with character ch added at the end
    return str + ch

def myCount( str, ch ):
    # Return the number of times character ch appears in str.
    counter = 0
    for letter in range(len(str)):
        if str[letter] == ch:
            counter += 1
    return counter

def myExtend( str1, str2 ):
    # Return a new string that contains the elements of
    # str1 followed by the elements of str2, in the same
    # order they appear in str2.
    str3 = str1 + str2
    return str3


def myMin( str ):
    # Return the character in str with the lowest ASCII code.
    if str == "":
        print("Empty string: no min value")
        return
    else:
        minimum = ord(str[0 : 1])
        for letter in str:
            asciiValue = int(ord(letter))
            if asciiValue < minimum:
                minimum = asciiValue
        return chr(minimum)


def myInsert( str, i, ch ):
    # Return a new string like str except that ch has been
    # inserted at the ith position. Print "Invalid index" if
    # i is greater than the length of str and return None.
    number = len(str)
    if i > number:
        print("Invalid index")
        return
    elif i == 0:
        return ch + str
    elif i == number:
        return str + ch
    elif 0 < i < number:
        return str[0 : i] + ch + str[ i : ]
```

```
def myPop( str, i ):
    # Return two results:
    # 1. a new string that is like str but with the ith
    #    element removed;
    # 2. the value that was removed.
    # Print "Invalid index" if i is greater than or
    # equal to len(str), and return str unchanged and None
    number = len(str)
    if i > number or i == len(str):
        print("Invalid index")
        return str, None
    elif i == "0":
        return str[1 : ] , str[ i : i+1 ]
    else:
         return str[0 : i] + str[ i + 1 : ] , str[ i : i + 1]

def myFind( str, ch ):
    # Return the index of the first (leftmost) occurrence of
    # ch in str, if any.  Return -1 if ch does not occur in str.
    if ch not in str:
        return -1
    elif ch in str:
        letterFound = str.index(ch)
        return letterFound

def myRFind( str, ch ):
    # Return the index of the last (rightmost) occurrence of
    # ch in str, if any.  Return -1 if ch does not occur in str.
    if ch not in str:
        return -1
    else:
        letterFound = -1
        for letter in range(0, len(str)):
            if str[letter] == ch:
                letterFound = letter
        return letterFound

def myRemove( str, ch ):
    # Return a new string with the first occurrence of ch
    # removed.  If there is none, return str.
    if ch not in str:
        return str
    elif ch in str:
        letter = str.index(ch)
        if str[letter] == ch:
            newString = str[0 : letter] + str[ letter + 1 : ]
        return newString


def myRemoveAll( str, ch ):
    # Return a new string with all occurrences of ch.
    # removed.  If there are none, return str.
    endString = ""
    if ch not in str:
        return str
    elif ch in str:
        for letter in str:
            if letter == ch:
                pass
            else:
                endString += letter
        return endString


def myReverse( str ):
    # Return a new string like str but with the characters
    # in the reverse order.
    return str[ : : -1]
```