```
/*
Filename: p3.cpp
Author(s): Zachary Rea and Parker Ross
Date: 5 February 2023
Description: The cpp for circular lists
*/
#include <iostream>  //allows for usage of cin, cout, and cerr

#include "p3.h"

using namespace std;

//********************************************************************************
//Constructors and De-constructors
//********************************************************************************
//Constructor
//Written by Zach

cStringList::cStringList(int capacity) {
    this->listCapacity = listCapacity;
    a = new string[listCapacity];
    listCount = 0;
    first = 0;
    last = 0;
}

//********************************************************************************
//De-constructor
//Written by Zach

cStringList::~cStringList() {
    delete [] a;
}

//********************************************************************************
//Private functions
//********************************************************************************
//Function for decrementing passed value
//Written by Parker

void cStringList::decVal(int &value){
    //prevents decVal from going beyond list capacity
    if (value > 0) {
        value = value - 1;
    } else {
        value = listCapacity - 1;
    }
}

//********************************************************************************
//Function for Incrementing passed value
//Written by Parker

void cStringList::incVal(int &value){
    //prevents incVal from going beyond list capacity
    if (value < listCapacity - 1) {
        value = value + 1;
    } else {
        value = 0;
    }
}

//********************************************************************************
//Public functions
//********************************************************************************
//Function for inserting a string at first position
//Written by Zach

bool cStringList::insert(string text) {
    bool rc = false;
    //Ensure there is space
    if (listCount < listCapacity) {
        if (listCount != 0) {
            decVal(first);
        }
        a[first] = text;
        listCount++;
        rc = true;
    }
    return rc;
}

//********************************************************************************
//Function for adding a string to the last position
//Written by Zach

bool cStringList::add(string text) {
    bool rc = false;
    if (listCount < listCapacity) {
        //move the last place
        incVal(last);
        a[last] = text;
        if (listCount == 0) {
            first = last;
        }
        listCount++;
        rc = true;
    }
    return rc;
}

//********************************************************************************
//Function for inserting a string at the given index
//Written by Zach

bool cStringList::insertAt(int index, string text) {
    bool rc = false;
    //Check for valid index
    if ((index <= listCount) && (index >= 0)) {
        incVal(last);
        int temp = last;
        //walk through the list
        for (int i = listCount; i > index; i--) {
            int move = temp;
            decVal (move);
            //copy the contents of the list down
            a[temp] = a[move];
            decVal(temp);
        }
        a[temp] = text;
        listCount++;
        rc = true;
    }
    return rc;
}

//********************************************************************************
//Funtion for deleting the contents of a given index
//Written by Zach

bool cStringList::deleteAt(int index, string &text) {
    bool rc = false;
    //check for valid index
    if ((index >= 0) && (index < listCount)) {
        int i = first;
        //walk through the list strarting from first
        while (i <= index) {
            //use incVal to ensure proper wrapping
            //this also allows us to use the index value
            incVal(i);
        }
        //set the text to the string that is being deleted
        text = a[i];
        //overwrite the rest of the contents
        while (i != last) {
            int temp = i;
            incVal(i);
            a[temp] = a[i];
        }
        decVal(last);
        listCount--;
        rc = true;
    }
    return rc;
}

//******************************************************************************
//Function for reading the value at given index
//written by Parker

bool cStringList::readAt(int index, string &text) {
    bool rc = false;
    //checking for index within the list
    if ((index >= 0) && (index < listCount)) {
        int read = first;
        int i = 0;
        //shifts entries to the left of index
        while (i < index) {
            i++;
            //use incVal to allow proper wrapping
            //this ensures that index is correct
            incVal(read);
        }
        text = a[read];
        rc = true;
    }
    return rc;
}

//******************************************************************************
//Function for deleting the first entry
//Written by Zach

bool cStringList::deleteFirst(string &text) {
    bool rc = false;
    if (listCount > 0) {
        //copy the text that is being removed
        text = a[first];
        //move the first indicator
        incVal(first);
        if (listCount == 1) {
            last = first = 0;
        }
        listCount--;
        rc = true;
    }
    return rc;
}

//******************************************************************************
//Function for deleting the last entry
//Written by Zach

bool cStringList::deleteLast(string &text) {
    bool rc = false;
    if (listCount > 0) {
        //copy the text that is being deleted
        text = a[last];
        //move the last indicator
        decVal(last);
        if (listCount == 1) {
            last = first = 0;
        }
        listCount--;
        rc = true;
    }
    return rc;
}

//******************************************************************************
//Fucntion for clearing list
//written by Parker

void cStringList::clear() {
    //empties list
    listCount = 0;
    first = last = 0;
}

//******************************************************************************
//function for returning size of list
//written by Parker

int cStringList::count() const{
    //returns number of list entries
    return listCount;
}

//******************************************************************************
//Function for returning the index of the given string
//Written by Zach

int cStringList::getIndex(string text) {
    int rc = -1;
    int i = first;
    //search for the text in a loop til last
    while ((a[i] != text) && (i != last)) {
        incVal(i);
    }
    //if found, record the index it was found
    if (a[i] == text) {
        rc = i;
    }
    return rc;
}

//******************************************************************************
//Function for printing values in circular list
//Written by Parker

void cStringList::printIt() const{
    if (listCount) {
        int print = first;
        int temp = last;
        if (temp < listCount) {
            temp++;
        } else {
            temp = 0;
        }
        int i = 0;
        while (print != temp){
            cout << "At pos " << i << " there is " << a[print] << "\n";
            if (print < listCount) {
                print++;
            } else {
                print = 0;
            }
            i++;
        }
    }
}
```