

UNIVERSIDAD NACIONAL SAN AGUSTIN DE AREQUIPA  
FACULTAD DE INGENIERÍAS  
ESCUELA PROFESIONAL DE CIENCIA DE LA COMPUTACIÓN



**LABORATORIO 03**

**Asignatura:** Ciencia de la Computación II

**Docente:** Rolando Jesus Cardenas Talavera

**Grupo:** B

**Estudiante:** Roni Ezequiel Montañez Pacco

**Arequipa-Perú**

**27 de abril del 2024**

# Ejercicios

1. Utilizando un puntero de caracteres (`char*`), cree una función que reciba dicho puntero y se le asigne contenido mediante el ingreso de texto por teclado. Cree una segunda función que transforme dicha cadena en Mayúsculas. Ambas funciones deben ser de tipo `void`, es decir, no retornan valores. Muestre el texto en mayúsculas desde la función principal.

```
) build/ejercicio1
Ingrese su cadena: Hello World!
HELLO WORLD!
```

## CÓDIGO

---

```
#include <iostream>
using namespace std;

void cadena(char *&punt){
    int tam=0;
    char *tmp;
    char c;
    punt = new char[100];
    cout << "Ingrese su cadena: ";
    cin.get(c);
    while(c!=char(10)){
        *(punt+tam) = c;
        cin.get(c);
        tam++;
    }
    tmp = new char[tam];
    for(int i=0;i<tam;i++){
        *(tmp+i) = *(punt+i);
    }
    delete [] punt;
    punt = tmp;
}

void mayuscula(char *punt){
    int it=0;
    do {
        if(*(punt+it) >= 'a' && *(punt+it) <= 'z'){
            *(punt+it)-=32;
        }
        it++;
    } while(*(punt+it)!='\0');
}

int main(){
    char *cad;
```

```

cadena(cad);
mayuscula(cad);
cout << cad << endl;
delete [] cad;
}

```

---

2. Implemente una función que reciba dos punteros de caracteres con contenido y que indique si el contenido del primer puntero es lexicográficamente mayor, menor o igual al contenido del segundo puntero. Retorne de la función un valor numérico que indique el resultado obtenido.

```

) build/ejercicio2
"Hola Mundo" es mayor lexicograficamente a "Hello World"

```

## CÓDIGO

---

```

#include <iostream>
using namespace std;

int comparar(char *cad1, char *cad2){
    int tam=0;
    while(*(cad1+tam) != '\0' && *(cad2+tam) != '\0'){
        if(*(cad1+tam) < *(cad2+tam)){
            return -1;
        }
        if(*(cad1+tam) > *(cad2+tam)){
            return 1;
        }
        tam++;
    }
    if(*(cad1+tam) != '\0'){
        return 1;
    }
    if(*(cad2+tam) != '\0'){
        return -1;
    }
    return 0;
}

int main(){
    char *cad1 = new char[10]{'H','o','l','a',' ','M','u','n','d','o'};
    char *cad2 = new char[11]{'H','e','l','l','o',' ','W','o','r','l','d'};
    //char *cad2 = new char[10]{'H','o','l','a',' ','M','u','n','d','o'};
    //char *cad1 = new char[12]{'H','e','l','l','o',' ','w','o','r','l','d','!'};

    int res = comparar(cad1,cad2);
    if(res == -1){
        cout << "\"" << cad1 << "\" es menor lexicograficamente a \"" << cad2 << "\"";
    }
}

```

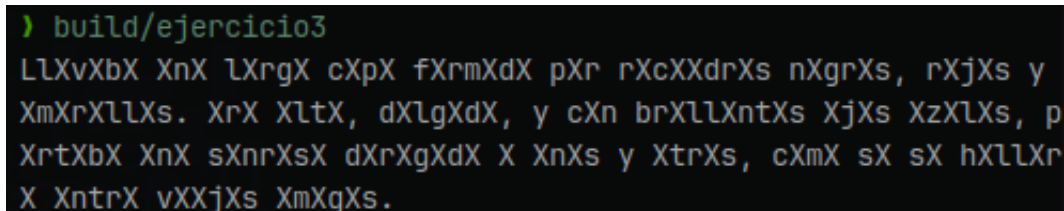
```

} else if(res == 1){
    cout << "\"" << cad1 << "\" es mayor lexicograficamente a \"" << cad2 << "\"";
} else {
    cout << "\"" << cad1 << "\" es igual lexicograficamente a " << cad2 << "\"";
}
cout << endl;
delete [] cad1;
delete [] cad2;
}

```

---

3. Cree una función que reciba un puntero de caracteres por referencia conteniendo un texto de 100 caracteres como mínimo. Utilizando aritmética de punteros, reemplace cada vocal con una X y muestre el resultado del contenido del puntero modificado.



```

) build/ejercicio3
LlXvXbX XnX lXrgX cXpX fXrmXdX pXr rXcXXdrXs nXgrXs, rXjXs y
MmXrXlXs. XrX XltX, dXlgXdX, y cXn brXlXntXs XjXs XzXlXs, p
XrtXbX XnX sXnrXsX dXrXgXdX X XnXs y XtrXs, cXmX sX sX hXlXr
X XntrX vXjXs XmXgXs.

```

## CÓDIGO

---

```

#include <iostream>
using namespace std;

void reemplazar(char *&punt){
    int it=0;
    char vocales[10] = {65,69,73,79,85,97,101,105,111,117};
    while ( *(punt+it) != '\0'){
        for(char voc : vocales) {
            if(*(punt+it) == voc){
                *(punt+it) = 'X';
            }
        }
        it++;
    }
}

int main(){
    char *texto = new char[205]{
        //Llevaba una larga capa formada por recuadros negros,
        //rojos y amarillos. Era alto, delgado y con brillantes
        //ojos azules, portaba una sonrisa dirigida a unos y otros,
        //como si se hallara entre viejos amigos.
        'L', 'l', 'e', 'v', 'a', 'b', 'a', ' ', 'u', 'n', 'a', ' ',
        'l', 'a', 'r', 'g', 'a', ' ', 'c', 'a', 'p', 'a', ' ', 'f',
        'o', 'r', 'm', 'a', 'd', 'a', ' ', 'p', 'o', 'r', ' ', 'r',

```

```

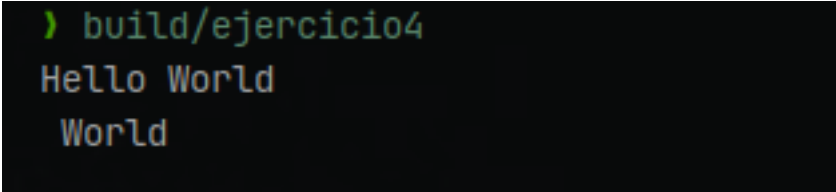
'e', 'c', 'u', 'a', 'd', 'r', 'o', 's', ' ', 'n', 'e', 'g',
'r', 'o', 's', ' ', 'r', 'o', 'j', 'o', 's', ' ', 'y',
' ', 'a', 'm', 'a', 'r', 'i', 'l', 'l', 'o', 's', ' ', 'E',
'r', 'a', ' ', 'a', 'l', 't', 'o', ' ', 'd', 'e',
'l', 'g', 'a', 'd', 'o', ' ', 'y', ' ', 'c', 'o', 'n',
' ', 'b', 'r', 'i', 'l', 'l', 'a', 'n', 't', 'e', 's', ' ',
'o', 'j', 'o', 's', ' ', 'a', 'z', 'u', 'l', 'e', 's', ' ',
' ', 'p', 'o', 'r', 't', 'a', 'b', 'a', ' ', 'u', 'n', 'a',
' ', 's', 'o', 'n', 'r', 'i', 's', 'a', ' ', 'd', 'i', 'r',
'i', 'g', 'i', 'd', 'a', ' ', 'a', ' ', 'u', 'n', 'o', 's',
' ', 'y', ' ', 'o', 't', 'r', 'o', 's', ' ', 'c', 'o',
'm', 'o', ' ', 's', 'i', ' ', 's', 'e', ' ', 'h', 'a', 'l',
'l', 'a', 'r', 'a', ' ', 'e', 'n', 't', 'r', 'e', ' ', 'v',
'i', 'e', 'j', 'o', 's', ' ', 'a', 'm', 'i', 'g', 'o', 's',
'.'
};

reemplazar(texto);
cout << texto << endl;
delete [] texto;
}

```

---

4. Implemente una función que reciba dos punteros de caracteres, realice el proceso de concatenación de ambas cadenas. Copie el contenido del segundo puntero al primero y muestre ambos punteros al finalizar el proceso.



```

) build/ejercicio4
Hello World
World

```

## CÓDIGO

---

```

#include <iostream>
using namespace std;

void concatenar(char *&cad1, char *cad2){
    int tam1=0,tam2=0;
    while(*(cad1+tam1) != '\0' || *(cad2+tam2) != '\0'){
        if(*(cad1+tam1) != '\0'){
            tam1++;
        }
        if(*(cad2+tam2) != '\0'){
            tam2++;
        }
    }
    char *tmp = new char[tam1+tam2];
    for(int i=0; i<tam1; i++){
        *(tmp+i) = *(cad1+i);
    }
}

```

```

    }
    for(int i=tam1; i<tam1+tam2; i++){
        *(tmp+i) = *(cad2+i-tam1);
    }
    delete [] cad1;
    cad1 = tmp;
}

int main(){
    char *cad1 = new char[5]{'H','e','l','l','o'};
    char *cad2 = new char[6]{' ','W','o','r','l','d'};
    concatenar(cad1,cad2);
    cout << cad1 << endl;
    cout << cad2 << endl;
    delete [] cad1;
    delete [] cad2;
}

```

---

5. Genere un ejemplo de dangling pointer con un puntero de caracteres.

```

) build/ejercicio5
zsh: segmentation fault (core dumped) build/ejercicio5

```

## CÓDIGO

---

```

#include <iostream>
using namespace std;

int main(){
    int *pointer;
    if(true){
        *pointer = 10;
    }
    cout << *pointer << endl;
}

```

---