

UNIVERSIDAD NACIONAL DEL ALTIPLANO DE **PUNO**

*Facultad de ingeniería mecánica eléctrica, electrónica y
sistemas*

Escuela profesional de ingeniería de sistemas



Trabajo en clase 1 y tabulaciones

PRESENTADO POR:

- Ccallo Arpita Jhak Esnayder

CURSO:

ALGORITMOS Y ESTRUCTURAS DE DATOS

DOCENTE:

ZANABRIA GALVEZ ALDO HERNAN

SEMESTRE IV

PUNO – PERÚ

2025

Actividad en clase y tabulaciones en papel

1. Diseñar un algoritmo que determine si una cadena es un palíndromo (sin usar funciones integradas).

Código:

```
1  #include <iostream>
2  using namespace std;
3
4  bool esPalindromo(string palabra){
5
6      int tamano=0;
7      bool palindromo = true;
8
9      for(char letra : palabra){
10         tamano++;
11     }
12
13     tamano-=1;
14
15     for(int i=0; i<(tamano)/2; i++){
16
17         if(palabra[i]!= palabra[tamano-i]){
18             palindromo=false;
19             break;
20         }
21     }
22
23     return palindromo;
24 }
25
26
27 string conMinuscula(string palabra){
28     int cont=0;
29     for(char letra: palabra){
30
31         if(letra>='A' && letra<='Z'){
32             palabra[cont]+=32;
33         }
34
35         cont++;
36     }
37
38     return palabra;
39 }
40
41
42 int main(){
43
44     string palabra;
45     cout<<"Ingrese una palabra: ";
46     cin>>palabra;
47
48     if(esPalindromo(conMinuscula(palabra))){
49         cout<<"la palabra es un palindromo"<<endl;
50     }
51     else{
52         cout<<"la palabra no es un palindromo"<<endl;
53     }
54
55     cout<<palabra;
56
57     return 0;
58 }
59
```

Código y tabulación en papel

En C++

Ejercicio 1

bool esPalindromo (string palabra) {

int tamano = 0;

bool palindromo = true;

for (char letra : palabra) {

tamano++;

}

tamano--;

for (int j = 0; j < tamano/2; j++) {

if (palabra[j] != palabra[tamano-j]) {

palindromo = false;

break;

}

}

return palindromo;

Palabra	tamano	Palindromo	letra	j	Resultado
1 2 3 2 1	0	true	-	-	→ true
1 2 3 2 1	1	true	'1'	-	
1 2 3 2 1	2	true	'2'	-	
1 2 3 2 1	3	true	'3'	-	
1 2 3 2 1	4	true	'2'	-	
1 2 3 2 1	5	true	'1'	-	
1 2 3 2 1	4	"	-	-	
1 2 3 2 1	4	true	-	0	
1 2 3 2 1	4	true	-	1	
1 2 3 2 1	4	true	-	1	

2. Elaborar un algoritmo que recorra un arreglo de N elementos y determine el segundo valor más alto sin ordenarlo.

Código:

```
1  #include <iostream>
2  #include <climits>
3  using namespace std;
4
5  int segMayor(int array[], int n){
6
7      int mayor=INT_MIN; //INT_MIN es una constante definida en <climits>
8      int indMayor;
9      for(int i=0; i<n; i++){
10         if(mayor<array[i]){
11             mayor=array[i];
12             indMayor=i;
13         }
14     }
15
16     mayor=INT_MIN;
17
18     for(int i=0; i<n; i++){
19         if(mayor<array[i] && array[i]!=array[indMayor]){
20             mayor=array[i];
21         }
22     }
23
24     return mayor;
25 }
26
27 int main(){
28     int n;
29     cout<<"introduzca el tamaño del array: ";
30     cin>>n;
31     int array[n];
32     cout<<"introduzca los datos del array: ";
33     for(int i=0; i<n; i++){
34         cin>>array[i];
35     }
36
37     cout<<"El segundo número mayor es: "<<segMayor(array,n);
38
39     return 0;
40 }
```

Código y Tabulación a mano

Ejercicio 2: [2, 4, 7, 1, 3] 5

```
int SegMayor (int array[], int n) {
```

```
    int mayor = INT_MIN;
```

```
    int indMayor;
```

```
    for (int i = 0; i < n; i++) {
```

```
        if (mayor < array[i]) {
```

```
            mayor = array[i];
```

```
            indMayor = i;
```

```
        }
```

```
    mayor = INT_MIN;
```

```
    for (int i = 0; i < n; i++) {
```

```
        if (mayor < array[i] && array[i] != array[indMayor]) {
```

```
            mayor = array[i];
```

```
        }
```

```
    }
```

```
    return mayor;
```

```
}
```

array	i	indMayor	mayor
[2, 4, 7, 1, 3]	0	0	2
[2, 4, 7, 1, 3]	1	1	4
[2, 4, 7, 1, 3]	2	2	7
[2, 4, 7, 1, 3]	3	2	7
[2, 4, 7, 1, 3]	4	2	7

array	i	indMayor	mayor
[2, 4, 7, 1, 3]	0	2	INT_MIN
[2, 4, 7, 1, 3]	1	2	2
[2, 4, 7, 1, 3]	2	2	4
[2, 4, 7, 1, 3]	3	2	4
[2, 4, 7, 1, 3]	4	2	4

SHARK
GENERATION

3. Simular una calculadora de tarifas para transporte público basada en el tipo de usuario y distancia recorrida.

Código:

```
1  #include <iostream>
2  using namespace std;
3
4  //el tipo de usuario define el valor base de la tarifa
5
6
7  double calTarifa(int t_usuario, int d_recorrida){
8      double tarifa;
9      //definimos la tarifa base
10     switch(t_usuario){
11         case 1:
12             tarifa=0.4;
13             break;
14         case 2:
15             tarifa=0.5;
16             break;
17         case 3:
18             tarifa=1.0;
19             break;
20     }
21
22     tarifa = tarifa + d_recorrida*1;
23
24     return tarifa;
25 }
26
27 int main(){
28
29     int t_usuario, d_recorrida;
30
31     cout<<"Indique el tipo de usuario: "<<endl;
32     cout<<"1. escolar "<<endl;
33     cout<<"2. universitario: "<<endl;
34     cout<<"3. adulto "<<endl;
35     cin>>t_usuario;
36
37     cout<<"Indique la distancia recorrida en km: "<<endl;
38     cin>>d_recorrida;
39
40     cout<<"la tarifa es de: "<<calTarifa(t_usuario,d_recorrida)<<"soles ";
41
42
43     return 0;
44 }
45
```


Código y tabulación a mano:

Ejercicio 3:

código:

```
double calTarifa (int t_usuario, int d_recorrida) {  
    double tarifa;  
    switch (t_usuario) {  
        case 1: tarifa = 0.4;  
            break;  
        case 2: tarifa = 0.5;  
            break;  
        case 3: tarifa = 1.0;  
            break;  
    }  
    tarifa = tarifa + d_recorrida * 0.5;  
    return tarifa;  
}
```

Tabulación

t_usuario	d_recorrida	tarifa
2	5	0.5
2	5	3