



UNIVERSIDAD NACIONAL DE SAN AGUSTIN

ESCUELA PROFESIONAL DE  
CIENCIA DE LA COMPUTACIÓN  
COMPILADORES

---

Practica 1

---

DATOS

***Alumnos :***

*Johann F. Huaypuna*

*Huanca*

***Profesor:***

*Carlos Atencio Torres*

## Índice

<b>1. Implementación con C++</b>	<b>2</b>
1.1. Pregunta 1 . . . . .	2
1.1.1. Código . . . . .	2
1.1.2. Resultado . . . . .	3
1.2. Pregunta 2 . . . . .	3
1.2.1. Código . . . . .	3
1.2.2. Resultado . . . . .	4
1.3. Pregunta 3 . . . . .	5
1.3.1. Código . . . . .	5
1.3.2. Resultado . . . . .	7
1.4. Pregunta 4 . . . . .	7
1.4.1. Código . . . . .	7
1.4.2. Resultado . . . . .	9

## 1. Implementación con C++

### 1.1. Pregunta 1

Leer toda una instrucción por consola (Ejemplo: "int temp;") y mostrar en pantalla letra por letra.

#### 1.1.1. Código

---


Código 1: Ejercicio 1

---

```
1 #include <iostream>
2 #include <string>
3 #include <algorithm>
4 #include <cctype>
5 #include <fstream>
6 #include <typeinfo>
7 using namespace std;
8 void letraXletra(){
9     string instruccion="";
10    cout<<"escribir instruccion: ";
11    getline(cin, instruccion);
12    int n=instruccion.size();
13    for (int i = 0; i < n; ++i){
14        cout<<instruccion[i]<<endl;
15    }
16 }
17 int main(int argc, char const *argv[]){
18     letraXletra();
19     return 0;
20 }
```

---

### 1.1.2. Resultado



```

PS D:\> g++ compi.cpp
PS D:\> .\a.exe
escribir instruccion: int temp=0;ac
1 void letraLetra(){
2     string instruccion="";
3     cout<<"escribir instruccion: ";
4     // cin>> instruccion;
5     getline(cin, instruccion);
6     int n=instruccion.size();
7     for (int i = 0; i < n; ++i)
8     {
9         cout<<"\t"<<instruccion[i]
10    }
11 }
PS D:\>

```

Figura 1: Ejercicio 1

## 1.2. Pregunta 2

Leer un archivo de texto plano (archivo con un pseudocódigo) y muestre en pantalla letra por letra.

### 1.2.1. Código

Código 2: Ejercicio 2

---

```

1 #include <iostream>
2 #include <string>
3 #include <algorithm>
4 #include <cctype>
5 #include <fstream>
6 #include <typeinfo>
7 using namespace std;
8 void leerArchivo(string fichero){
9     fstream ficheroEntrada;
10    char letra;
11    ficheroEntrada.open ( fichero , ios::in );
12    if ( ficheroEntrada.is_open() ) {
13        while (!ficheroEntrada.eof()) {
14            ficheroEntrada >> letra;
15            cout<< letra<<" ";
16        }
17    ficheroEntrada.close();
18    }
19    else cout << "Fichero inexistente" << endl;

```

```

20 }
21 int main(int argc, char const *argv[]) {
22     leerArchivo("a.txt");
23     return 0;
24 }

```

---

El archivo con pseudocódigo es .a.txt

Código 3: a.txt

---

```

1 #include <iostream>
2 using namespace std;
3 void letraXletra() {
4     string instruccion="";
5     cout<<"escribir instruccion: ";
6     cin>> instruccion;
7     getline(cin, instruccion);
8     int n=instruccion.size();
9     for (int i = 0; i < n; ++i)
10    {
11        cout<<"\t"<<instruccion[i]<<endl;
12    }
13
14 }

```

---

### 1.2.2. Resultado

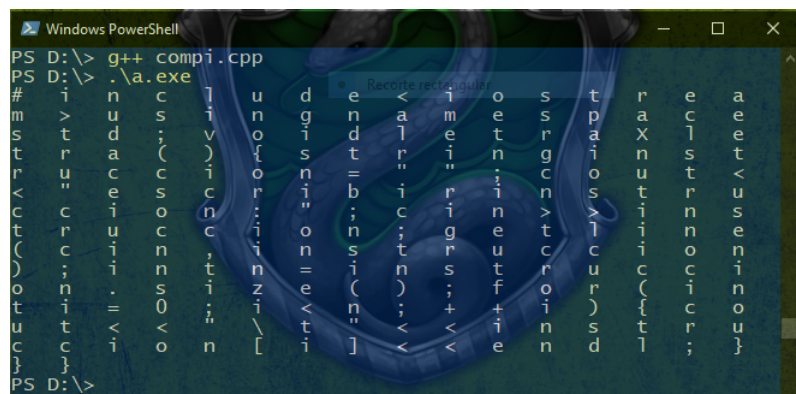


Figura 2: ejercicio 2

### 1.3. Pregunta 3

Crear un programa que cifre un pseudocódigo y otro que lo descifre, según la técnica del cifrado de César. El pseudocódigo se encuentra en un archivo de texto plano. El cifrado de César consiste en mover cada letra un determinado número de espacios en el alfabeto (puede ser 3 espacios)

#### 1.3.1. Código

---

Código 4: Contrast Stretching

---

```
1
2 #include <iostream>
3 #include <string>
4 #include <algorithm>
5 #include <cctype>
6 #include <fstream>
7 #include <typeinfo>
8 #include <ctype.h>
9 #include <vector>
10 #include <stdio.h>
11
12
13 using namespace std;
14 void cifradoCesar(string fichero ,int posiciones){
15     ifstream ifs(fichero);
16     string s;
17     string salida="";
18     vector<int> v;
19     ofstream file;
20     file.open("encriptado.txt");
21     while(getline(ifs,s)){
22         salida=salida+s;
23     }
24     int n=salida.size();
25     for (int i = 0; i < n; ++i){
26         v.push_back(toascii(salida[i]) + posiciones);
27     }
```

```
28     string encriptado="";
29     for (int j = 0; j < v.size(); ++j){
30         file<<char(v[j]);
31     }
32     file.close();
33 }
34
35 void descifradoCesar(string fichero ,int posiciones){
36     ifstream ifs(fichero);
37     string s;
38     string salida="";
39     vector<int> v;
40     ofstream file;
41     file.open("DesEncriptado.txt");
42     while(getline(ifs,s)){
43         salida=salida+s;
44     }
45     int n=salida.size();
46     for (int i = 0; i < n; ++i){
47         v.push_back(toascii(salida[i]) - posiciones);
48     }
49     string encriptado="";
50     for (int j = 0; j < v.size(); ++j){
51         encriptado=encriptado+ char(v[j]);
52         file<<char(v[j]);
53     }
54     file.close();
55 }
56
57 int main(){
58     cifradoCesar("a.txt",3);
59     descifradoCesar("Encriptado.txt",3);
60
61     return 0;
62 }
```

---

### 1.3.2. Resultado

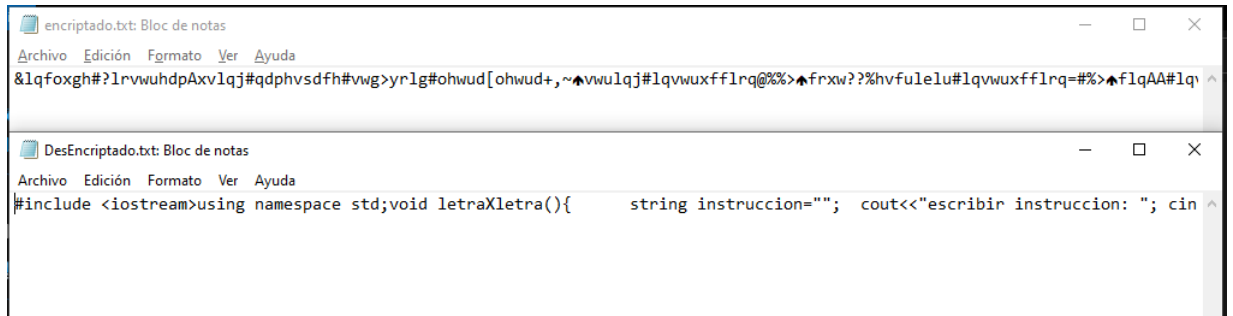


Figura 3: Ejercicio 3

### 1.4. Pregunta 4

Leer un archivo de texto plano, letra por letra hasta encontrar un separador (espacio, tabulador o salto de línea) y mostrar en pantalla si el vocablo se trata de un número entero, una palabra o un caracter especial (“+”, “-”, “\*”, “/”)

#### 1.4.1. Código

Código 5: Ejercicio 4

---

```
1 #include <iostream>
2 #include <string>
3 #include <algorithm>
4 #include <cctype>
5 #include <fstream>
6 #include <typeinfo>
7 using namespace std;
8
9 bool is_number(const std::string& s)
10 {
11     std::string::const_iterator it = s.begin();
12     while (it != s.end() && std::isdigit(*it)) ++it;
13     return !s.empty() && it == s.end();
14 }
15
16 void separadorEnLinea(string archivo){
17     string texto;
```



```
18     ifstream ifs(archivo);
19     string s;
20     string salida="";
21     //file.open(archivo);
22     while(getline(ifs,s)){
23         salida=salida+s;
24     }
25     texto=salida;
26
27     string aux;
28     char letra;
29     int n=texto.size();
30     for (int i = 0; i < n; ++i){
31         letra=texto[i];
32         if(letra!=' '){ //poner los separadores
33             aux=aux+letra;
34         }
35         else{
36             if (is_number(aux)){
37                 cout<<"\t"<<"entero"<<endl; //poner los tipos
38             }
39             else if (aux=="+" or aux=="-" or aux=="*" or aux=="/"){
40                 cout<<"\t"<<"Caracter Especial"<<endl;
41             }
42             else{
43                 cout<<"\t"<<"Palabra"<<endl;
44             }
45             aux="";
46         }
47     }
48 }
49
50 int main(int argc, char const *argv[])
51 {
52     separadorEnLine("a.txt");
53     return 0;
54 }
```

---

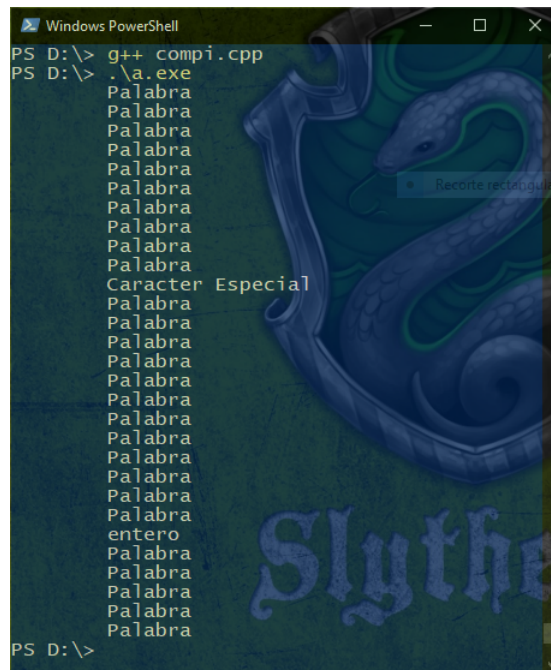


Figura 4: Ejercicio 4