

Actividad | 3 | Calculo de RFC

Lenguajes de Programación 1

Ingeniería en Desarrollo de Software



TUTOR: Francisco Ortega

ALUMNO: Diana Susana López Moreno

FECHA: Julio de 2023

Índice

Introducción	3
Descripción	4
Justificación	5
Desarrollo:	6
○ Codificación	6
○ Prueba del sistema	8
Conclusión	9
Referencias	10

Introducción

Como anteriormente hemos dicho el lenguaje C++ es un sistema de códigos orientado a objetos, estos nos ayudan a generar algoritmos que nos ayudan a resolver problemas cotidianos que van desde una operación simple matemática hasta todo un proceso de generación de venta en algún negocio.

El uso de este lenguaje nos ayuda a exponer códigos que siguen una serie de instrucciones y nos arrojan un resultado en base a ello, es decir creas un programa que te ayude a resolver desde una suma hasta algo más elaborado como un RFC como es el caso de esta actividad.

Con el uso de este lenguaje llega el momento en el que generar códigos resulta algo rápido y sencillo pues al estar usándolos constantemente es más fácil saber que código es el adecuado para la situación que se está realizando y nos lleva al resultado de forma mas rápida.

Descripción

Para esta actividad se necesita crear un programa que permita calcular el RFC de los nuevos empleados de la constructora AMC. Este debe generarse a partir de la captura de nombre, apellido paterno, apellido materno y fecha de nacimiento.

Actividad:

Con base en las siguientes reglas (en esta aplicación se deberán de usar los conceptos de Objetos y Clases, vistos en la materia), realizar una aplicación que permita capturar el RFC de los usuarios. Esta debe realizarse con sus respectivos requerimientos en lenguaje C++:

Ejemplo

RFC:	V	E	C	J	8	8	0	3	2	6	X	X	X
Posición	1	2	3	4	5	6	7	8	9	10	11	12	13

Se utilizará la plataforma que se ha venido utilizando desde el inicio de esta materia: https://www.onlinegdb.com/online_c_compiler), y se anexaran las capturas correspondientes del proceso hasta llegar a la generación de RFC.

Justificación

En esta actividad este código se creó con la finalidad de que se genere el RFC a partir de ingresar nombre, apellido paterno, apellido materno y fecha de nacimiento.

Esto nos sirve para tener acceso a una base de datos de forma rápida y así localizar al elemento que estas buscando con solo ingresar unos cuantos datos.

Después de realizar la actividad, la cual se complicó un poco, llegué a la conclusión de que como lo he dicho anteriormente si se conocen los códigos correctos es más fácil llegar al resultado esperado.

El link de GitHub se adjunta en el apartado de referencias al igual que en las otras actividades, se le da clic y te lleva a la página en donde se desarrolla el código, para que se valide que es un código funcional.

Algo adicional es que este RFC es sin homoclave ya que esta es generada por las dependencias de gobierno.

Desarrollo:

El proceso se llevó a cabo de la siguiente forma:

○ *Codificación*

```
#include <iostream>
```

```
#include <string>
```

```
class EmpleadoAMC {
```

```
private:
```

```
    std::string nombre;
```

```
    std::string apellidopaterno;
```

```
    std::string apellidomaterno;
```

```
    std::string fechadenacimiento;
```

```
public:
```

```
    void capturarDatos() {
```

```
        std::cout << "INGRESA NOMBRE: ";
```

```
        std::getline(std::cin >> std::ws, nombre);
```

```
        std::cout << "INGRESA APELLIDO PATERNO: ";
```

```
        std::getline(std::cin >> std::ws, apellidopaterno);
```

```
        std::cout << "INGRESA APELLIDO MATERNO: ";
```

```
        std::getline(std::cin >> std::ws, apellidomaterno);
```

```
        std::cout << "INGRESA FECHA DE NACIMIENTO (DD/MM/AAAA): ";
```

```
        std::getline(std::cin >> std::ws, fechadenacimiento);
```

```
    }
```

```

std::string generarRFC() {

    std::string rfc;

    //Primeros 2 caracteres

    rfc += apellidopaterno[0];

    rfc += apellidopaterno[1];

    for (char c : apellidopaterno) {

        if (c == 'A' || c == 'E' || c == 'I' || c == 'O' || c == 'U'){

            rfc += c;

            break;

        }

    }

    //3ra posición

    if (apellidomaterno.empty()) {

        rfc += 'X';

    } else {

        rfc += apellidomaterno[0];

    }

    //4ta posición

    rfc += (nombre[0] == '#' ? 'X' : nombre[0]);

    //5ta y 6ta posición (año de nacimiento)

    rfc += fechadenacimiento.substr(8, 2);

    //7ma y 8va posición (mes de nacimiento)

```

```

    rfc += fechadenacimiento.substr(3, 2);

    //9na y 10ma posición (día de nacimiento)

    rfc += fechadenacimiento.substr(0, 2);

    return rfc;
}

};

int main () {

    EmpleadoAMC EmpleadoAMC;

    EmpleadoAMC.capturarDatos();

    std::string rfc = EmpleadoAMC.generarRFC();

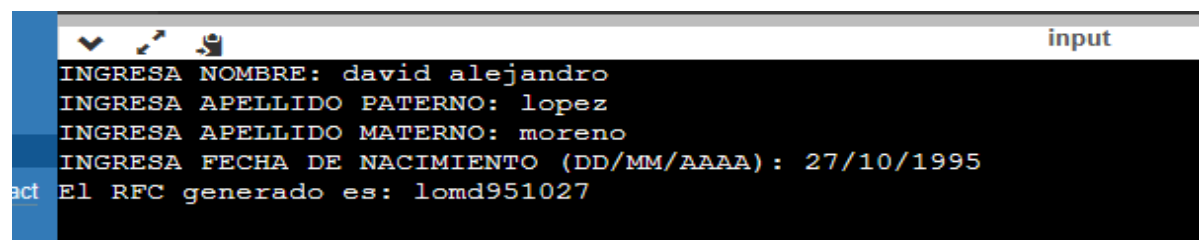
    std::cout << "El RFC generado es: " << rfc << std::endl;

    return 0;

}

```

○ *Prueba del sistema con nombre y dos apellidos:*

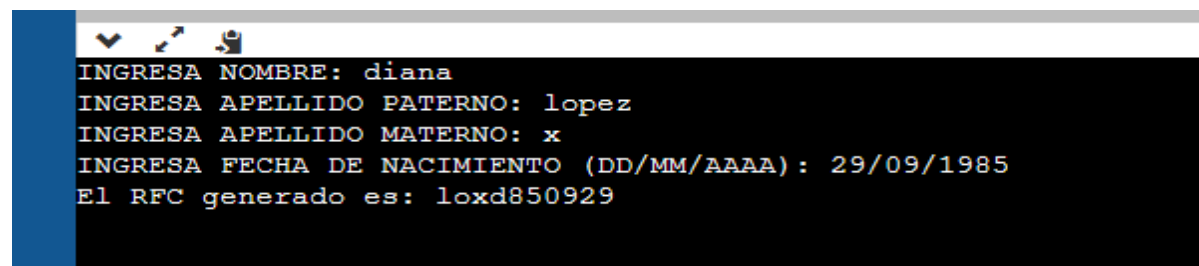


```

input
INGRESA NOMBRE: david alejandro
INGRESA APELLIDO PATERNO: lopez
INGRESA APELLIDO MATERNO: moreno
INGRESA FECHA DE NACIMIENTO (DD/MM/AAAA): 27/10/1995
El RFC generado es: lomd951027

```

○ *Prueba del sistema con nombre y un apellido:*



```

INGRESA NOMBRE: diana
INGRESA APELLIDO PATERNO: lopez
INGRESA APELLIDO MATERNO: x
INGRESA FECHA DE NACIMIENTO (DD/MM/AAAA): 29/09/1985
El RFC generado es: loxd850929

```


Conclusión

Esta actividad me llevo as tiempo del que tenia pensado ya que me costo lograr que se acomodaran los primeras dos letras del rfc hasta que vi el video unas tres veces, y después de bastantes intentos y varias horas de probar distintos comandos logré realizar el acomodo correcto de RFC.

Con esta actividad me di cuenta de qué incluso lo que se ve más sencillo lleva muchísimo trabajo detrás, no es solo que arroje las letras o los números, debe tener el acomodo correcto para que sea el resultado esperado y no termine arrojando información errónea, ya que esto puede afectar por ejemplo la base de datos de alguna empresa o algún registro de gobierno.

La conclusión sigue siendo la misma para mi entre más utilices el lenguaje C, más se te facilita la realización de los algoritmos con los códigos correctos, lo cual te permitirá llegar en menos tiempo al resultado requerido.

Se anexará el link para la visualización en GitHub:

<https://github.com/d1l0p/ldp1.git>

Y para validar el código:

<https://onlinegdb.com/z2xiFpypw>

Referencias

https://umi.edu.mx/coppel/IDS/plataforma/plan_estudios.php

COP L LP1 TU