Universidad Autónoma de Querétaro

Facultad de Informática

C.U. a 27/02/2023

Estructuras de datos

Nombre del Alumno: Rafael Barcena González

No. de Expediente: 307012

Correo electrónico: rbarcena14@alumnos.uaq.mx

Teléfono: 4424903125

Mini Proyecto Unidad 1: Juego de adivinanza de números

Este es un juego simple que te reta a adivinar un número secreto generado al azar por la computadora. Utilizaremos el lenguaje de programación C++ para desarrollar este juego.

El usuario ingresará un número y la computadora le dirá si el número ingresado es mayor o menor que el número secreto. El usuario tendrá un número limitado de intentos para adivinar el número correcto (5). Si el usuario adivina el número correcto dentro del número de intentos permitido, ¡ganará el juego! De lo contrario, perderá y se mostrará el número secreto.

El juego hace uso de los siguientes conceptos:

1.1. Estructura del lenguaje (sintaxis):

- Declaración de variables
- Operadores aritméticos
- Estructuras de control: if, else, while

1.2. Entrada/Salida:

- Salida por consola: cout
- Entrada por consola: cin

1.3. Estructuras de control:

• if, else: para verificar si el número ingresado por el usuario es correcto • while: para permitir al usuario adivinar hasta que lo haga correctamente o se quede sin intentos

1.4. Apuntadores:

• Para generar un número aleatorio, se utilizará un apuntador para almacenar la dirección de memoria de la variable que contiene el número generado.

rand()

Funcionalidades extra:

• Agregar una interfaz gráfica de usuario (GUI) para hacer la entrada/salida más atractiva y fácil de usar.

• Permitir al usuario elegir el rango de los números aleatorios generados.

VI. Contenido temático
Unidad 1. Utilizando C++ **Competencias a desarrollar.** Construye software para diferentes tipos de aplicaciones, analizando el alcance del lenguaje C++, en el desarrollo de software. **Objetivo de la unidad.** Introducir al estudiante al lenguaje de programación C++ y uso de memoria dinámica a través del tipo de dato apuntador. **Temas** 1.1. Estructura del lenguaje (sintaxis) 1.2. Entrada/Salida 1.3. Estructuras de control 1.4. Apuntadores

Explicación del código del mini proyecto 1

Código:

```
#include <stdlib.h>
#include <iostream>
#include <time.h>

using namespace std;
int main()
{
    int N, random, cont = 1, max, min;
    srand(time(NULL));
    cout<<"Tiene que adivinar un numero\n";
    cout<<"Elige el numero del limite menor: \n";
    cin>>min;
    cout<<"Elige el numero del limite mayor: \n";
    cin>>max;

    random = rand()% max + min;
    do
    {
        cout<<"elija un numero: "; cin>>N;
        if (N == random)
        {
            cout<<"\nAdivinaste el numero\n";
            cout<<"Numero de intentos: "<<cont + 1<<endl;
            abort();
        }
        if (cont == 5)
        {
            cout<<"\nEl numero que buscabas era " << random << "\n";
            abort();
        }

        if (N > random){
            cout<<"\nel numero es menor, elija otro \n";
        }
        else if (N <random){
            cout<<"\nel numero es mayor, elija otro \n";
        }
        cont++;
    }while (N !=random);

    return 0;
}
```

Primero exportamos las siguientes librerías:
#include <stdlib.h> = Para generar un numero aleatorio
#include <iostream> = para la entrada y salida de datos
#include <time.h> = para que funcione el srand(time(NULL));

La función srand() se encarga de establecer el valor inicial para el generador de números pseudoaleatorios utilizado por rand(). Si no se llama a srand() para establecer una semilla, el generador de números pseudoaleatorios se inicializará con un valor predeterminado, lo que significa que los números pseudoaleatorios generados por rand() serán los mismos en cada ejecución del programa.

Al utilizar la función srand(time(NULL)), se establece la semilla inicial del generador de números pseudoaleatorios utilizando el tiempo actual del sistema. De esta manera, se garantiza que la secuencia de números generados por rand() será diferente cada vez que se ejecute el programa.

Primero declaramos e inicializamos las variables
En segundo lugar, pedimos al usuario que ingrese los limites para generar el numero aleatorio a adivinar del juego y mandamos a llamar el método srand(time(NULL)), después, declaramos la variable random y la inicializamos con un número random que estará entre los limites dados

En tercer lugar, iniciamos un do while en donde se agregaran varios IF para determinar si el numero dado por el usuario es igual al generado por la consola, si no, indicara si e numero que busca es mayor o menor, además, si adivina el numero correcto en el limite de intentos ganaras y si no, mostrara un mensaje que diga el numero esperado.