Manejo de tipos de datos en Binario

Double < > bits

codigo para ver como se registras números double en binarios de 64 bits

bitsDouble.cpp

```
#include <iostream>
#include <bitset>
#include <cstdint>
#include <string>
#include <sstream>

void mostrarBits(double numero) {
   union {
       double input;
       uint64_t output;
   } data;
   data.input = numero;

   // Mostrar todos los bits
   std::cout << "Todos los bits: "<<
"signo(1)|exponente(11)|mantisa(52):\nseeeeeeeeee.emmmmmmmmm.mmmmmmmmm.mmmmmmmm
mm.mmmmmmmmm.mmmmmmmmm.mm\n" << std::bitset<64>(data.output) << std::endl;

   // Bits de signo
   std::cout << "Bits de signo: " << std::bitset<1>(data.output >> 63) <<
std::endl;

   // Bits de exponente
   std::cout << "Bits de exponente: " << std::bitset<11>((data.output >> 52)
& 0x7FF) << std::endl;

   // Bits de mantisa
   std::cout << "Bits de mantisa: " << std::bitset<52>(data.output &
0xFFFFFFFFFFFFF) << std::endl;
}
```

```cpp
double binarioADouble(const std::string& binario) {
    // Convertir la cadena binaria a un número double
    std::bitset<64> bits(binario);
    union {
        uint64_t input;
        double output;
    } data;
    data.input = bits.to_ullong();
    return data.output;
}

int main() {
    char opcion;
    std::cout << "¿Desea ingresar el número en formato double (d) o binario (b)? ";
    std::cin >> opcion;

    if (opcion == 'd') {
        double numero;
        std::cout << "Ingrese un número double: ";
        std::cin >> numero;
        mostrarBits(numero);
    } else if (opcion == 'b') {
        std::string binario;
        std::cout << "Ingrese el número en formato binario (64 bits) :: signo(1)|exponente(11)|mantisa(52):\nseeeeeeeee.emmmmmmmmm.mmmmmmmmm.mmmmmmmmm.mmmmmmmmm.mmmmmmmmm.mm \n";
        std::cin >> binario;
        double numero = binarioADouble(binario);
        std::cout << "El número double correspondiente es: " << numero << std::endl;
    } else {
        std::cout << "Opción no válida." << std::endl;
    }

    return 0;
}
```

```
edi@fedora:~/docencia/c++$ g++ bitsDouble.cpp -o bd
edi@fedora:~/docencia/c++$ ./bd
¿Desea ingresar el número en formato double (d) o binario (b)? d
Ingrese un número double: 3.14
Todos los bits: signo(1) | exponente(11) | mantisa(52):
s eeeeeeeeeee mmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmm
0 10000000000 1001000111101011100001010001111010111000010100011111
Bits de signo: 0
Bits de exponente: 10000000000
Bits de mantisa: 1001000111101011100001010001111010111000010100011111
```

Integer < > bits

bitsInteger.cpp

```
#include <iostream>
#include <bitset>
#include <cstdint>
#include <string>
#include <sstream>

void mostrarBits(int numero) {
   std::cout << "Bits: \n.|.....|.....|.....|.....|\n" <<
std::bitset<sizeof(int) * 8>(numero) << std::endl;
}

std::string enteroABinario(int numero) {
   std::stringstream ss;
   ss << std::bitset<sizeof(int) * 8>(numero);
   return ss.str();
}

int binarioAEntero(const std::string& binario) {
   return std::bitset<sizeof(int) * 8>(binario).to_ulong();
}
```

```cpp
int main() {
    char opcion;
    std::cout << "¿Desea ingresar el número en formato entero (e) o binario (b)? ";
    std::cin >> opcion;

    if (opcion == 'e') {
        int numero;
        std::cout << "Ingrese un número entero: ";
        std::cin >> numero;
        mostrarBits(numero);
    } else if (opcion == 'b') {
        std::string binario;
        std::cout << "Ingrese el número en formato binario: \n.|........|........|........|\n";
        std::cin >> binario;
        int numero = binarioAEntero(binario);
        std::cout << "El número entero correspondiente es: " << numero << std::endl;
    } else {
        std::cout << "Opción no válida." << std::endl;
    }

    return 0;
}
```

```
edi@fedora:~/docencia/c++$ g++ bitsInteger.cpp -o bi
edi@fedora:~/docencia/c++$ ./bi
¿Desea ingresar el número en formato entero (e) o binario (b)? e
Ingrese un número entero: 321
Bits:
.|........|........|........|
00000000000000000000000101000001
```

texto < > bits

bitsTexto.cpp

```cpp
#include <iostream>
#include <bitset>
#include <string>

void mostrarBitsPorLetras(const std::string& texto) {
    for (char c : texto) {
        std::bitset<8> bits(c);
        std::cout << bits << " ";
    }
    std::cout << std::endl;
}

std::string textoDesdeBits(const std::string& bits) {
    std::string texto;
    for (size_t i = 0; i < bits.size(); i += 8) {
        std::bitset<8> byte(bits.substr(i, 8));
        texto += static_cast<char>(byte.to_ulong());
    }
    return texto;
}

int main() {
    char opcion;
    std::cout << "¿Desea ingresar un texto (t) o una serie de bits (b)? ";
    std::cin >> opcion;

    if (opcion == 't') {
        std::string texto;
        std::cout << "Ingrese un texto: ";
        std::cin.ignore(); // Ignorar el salto de línea después de la entrada
anterior
        std::getline(std::cin, texto);

        std::cout << "Representación en bits de cada letra:" << std::endl;
        mostrarBitsPorLetras(texto);
```

```
    } else if (opcion == 'b') {
        std::string bits;
        std::cout << "Ingrese una serie de bits: ";
        std::cin >> bits;

        std::string texto = textoDesdeBits(bits);
        std::cout << "Texto correspondiente: " << texto << std::endl;
    } else {
        std::cout << "Opción no válida." << std::endl;
    }

    return 0;
}
```

```
edi@fedora:~/docencia/c++$ g++ bitsTexto.cpp -o bt
edi@fedora:~/docencia/c++$ ./bt
¿Desea ingresar un texto (t) o una serie de bits (b)? b
Ingrese una serie de bits: 01001000011011110110110001100001
Texto correspondiente: Hola
```

Conversor Binario

programa para escribir un texto y crear un archivo binario

conversorBinario.cpp

```
#include <iostream>
#include <fstream>
#include <string>
#include <vector>
#include <bitset>

class AsciiConverter {
public:
    AsciiConverter(const std::string& text) : text(text) {}
```

```
    std::vector<unsigned char> convertToBytes() const {
        std::vector<unsigned char> bytes;
        for (char c : text) {
            bytes.push_back(static_cast<unsigned char>(c));
        }
        return bytes;
    }

    void printBytes(const std::vector<unsigned char>& bytes) {
        std::cout << "Bytes: ";
        for (unsigned char byte : bytes) {
            std::cout << static_cast<int>(byte) << " ";
        }
        std::cout << std::endl;
    }

    void printBits(const std::vector<unsigned char>& bytes) {
        std::cout << "Bits: ";
        for (unsigned char byte : bytes) {
            std::bitset<8> bits(byte);
            std::cout << bits << " ";
        }
        std::cout << std::endl;
    }

private:
    std::string text; // Miembro de datos para almacenar el texto
};

int main(int argc, char *argv[]) {
    if (argc != 2) {
        std::cerr << "Uso: " << argv[0] << " <nombre_archivo>" << std::endl;
        return 1;
    }

    std::string fileName = argv[1];
    std::cout << "Ingrese el texto: ";
    std::string inputText;
    std::getline(std::cin, inputText);
```

```
    AsciiConverter converter(inputText);
    std::vector<unsigned char> bytes = converter.convertToBytes();

    std::ofstream outputFile(fileName, std::ios::binary);
    if (!outputFile.is_open()) {
        std::cerr << "Error: No se pudo abrir el archivo de salida." <<
std::endl;
        return 1;
    }

    converter.printBits(bytes);
    outputFile.write(reinterpret_cast<const char*>(bytes.data()),
bytes.size());
    outputFile.close();

    std::cout << "Cadena de bytes generada y guardada en '" << fileName <<
"'." << std::endl;

    return 0;
}
```

```
edi@fedora:~/docencia/c++$ g++ conversorBinario.cpp -o cbb
edi@fedora:~/docencia/c++$ ./cbb edi.bin
Ingrese el texto: Hola Jedi
Bits: 01001000 01101111 01101100 01100001 00100000 01001010 01100101 01100100 01101001
Cadena de bytes generada y guardada en 'edi.bin'.
```