Ödev 5 - Sayaç Sorusu Çözümü

Grup Üyeleri : Barış S. Yakar - Ceren Yanık - Fethi Omur - İbrahim F. Kayan

1. Çözüm

```
#include <iostream>
#include <string>
#include <map>

std::string binaryToHex(const std::string& binary) {
    std::map<std::string, char> binaryToHexMap;
    binaryToHexMap.insert(std::make_pair("0000", '0'));
    binaryToHexMap.insert(std::make_pair("0001", '1'));
    binaryToHexMap.insert(std::make_pair("0010", '2'));
    binaryToHexMap.insert(std::make_pair("0011", '3'));
    binaryToHexMap.insert(std::make_pair("0100", '4'));
    binaryToHexMap.insert(std::make_pair("0101", '5'));
    binaryToHexMap.insert(std::make_pair("0110", '6'));
    binaryToHexMap.insert(std::make_pair("0111", '7'));
    binaryToHexMap.insert(std::make_pair("1000", '8'));
    binaryToHexMap.insert(std::make_pair("1001", '9'));
    binaryToHexMap.insert(std::make_pair("1010", 'A'));
    binaryToHexMap.insert(std::make_pair("1011", 'B'));
    binaryToHexMap.insert(std::make_pair("1100", 'C'));
    binaryToHexMap.insert(std::make_pair("1101", 'D'));
    binaryToHexMap.insert(std::make_pair("1110", 'E'));
    binaryToHexMap.insert(std::make_pair("1111", 'F'));

    std::string hex;
    for (size_t i = 0; i < binary.size(); i += 4) {
        std::string chunk = binary.substr(i, 4);
        hex += binaryToHexMap[chunk];
    }
    return hex;
}

int main() {
    std::string binaryDigits[] = {
        "11101110", "00100100", "10111010", "10110110", "01110100",
        "11010110", "11011110", "10100100", "11111110", "11110110"
    };

    std::cout << "Enter the number of seconds: ";
    int seconds;
    std::cin >> seconds;

    int totalSeconds = seconds % 86400;
    int hour = (totalSeconds / 3600);
    int minute = (totalSeconds % 3600) / 60;
```

```
    std::string binaryHour = binaryDigits[hour / 10] + binaryDigits[hour % 10];
    std::string binaryMinute = binaryDigits[minute / 10] + binaryDigits[minute % 10];

    std::string hexHour = binaryToHex(binaryHour);
    std::string hexMinute = binaryToHex(binaryMinute);

    std::cout << "Hour Hex: " << hexHour << std::endl;
    std::cout << "Minute Hex: " << hexMinute << std::endl;

    return 0;
}
```

2. Çözüm

```
#include <iostream>
#include <vector>
#include <string>
#include <chrono>
#include <thread>

std::vector<std::string> sevenSegmentEncoding;

void initializeSevenSegmentEncoding() {
    sevenSegmentEncoding.push_back("11101110"); // 0
    sevenSegmentEncoding.push_back("00100100"); // 1
    sevenSegmentEncoding.push_back("10111010"); // 2
    sevenSegmentEncoding.push_back("10110110"); // 3
    sevenSegmentEncoding.push_back("01110100"); // 4
    sevenSegmentEncoding.push_back("11010110"); // 5
    sevenSegmentEncoding.push_back("11011110"); // 6
    sevenSegmentEncoding.push_back("10100100"); // 7
    sevenSegmentEncoding.push_back("11111110"); // 8
    sevenSegmentEncoding.push_back("11110110"); // 9
}

void displayDigit(int digit) {
    if (digit < 0 || digit > 9) {
        std::cout << "Invalid digit: " << digit << std::endl;
        return;
    }

    std::string encoding = sevenSegmentEncoding[digit];
    std::cout << encoding << "   " << digit << "  ";

    std::cout << std::endl;
}

void displayTime() {
    std::time_t currentTime = std::time(nullptr);
    std::tm* timeInfo = std::localtime(&currentTime);
```

```
    int hours = timeInfo->tm_hour;
    int minutes = timeInfo->tm_min;
    int seconds = timeInfo->tm_sec;

    std::cout << "     Clock     " << std::endl;
    std::cout << "===============" << std::endl;
    displayDigit(hours / 10);
    displayDigit(hours % 10);
    std::cout << "       " << std::endl;
    displayDigit(minutes / 10);
    displayDigit(minutes % 10);
    std::cout << "       " << std::endl;
    displayDigit(seconds / 10);
    displayDigit(seconds % 10);
}

int main() {
    initializeSevenSegmentEncoding();

    while (true) {
        displayTime();
        std::this_thread::sleep_for(std::chrono::seconds(1));
        // Clear the console (works on some systems)
        std::cout << "\033[2J\033[1;1H";
    }

    return 0;
}
```