Password decryptor documentation

**General description**

The program implements a method of decrypting passwords encrypted with MD5 using two approaches: **serial** and **parallel**. The duration of the two approaches is measured, so we can compare the efficiency in terms of execution time.

**Program structure**

The program is divided into the following classes:

1. **Decryptor**
2. **ParallelDecryptor**
3. **SerialDecryptor**
4. **MainProgram**

**1. Decryptor Class**

This class implements the methods for decrypting a password. It uses backtracking to generate and verify every possible password. Also called brute forcing.

**Properties:**

1. characters: a string containing all possible symbols from which a password can be formed;
2. password: an array of characters, used to store passwords;

**Methods:**

1. **bruteForceDecryption**
   1. Backtracking to generate all combinations of 4 characters in length and verify that their MD5 hash matches the encrypted hash (encryptedPass).
   2. If it finds the correct password, it displays the result and stops the thread using tokenSource.Cancel().
2. **CreateMD5Hash**
   1. Creates an MD5 hash for a password
   2. Returns the hash as a string

**2. ParallelDecryptor Class**

This class implements parallel decryption by creating a thread for each password. Each thread runs an instance of the Decryptor class.

**Properties:**

1. threads: an array of thread-uri
2. tokenSources: An array of CancellationTokenSource used to terminate each thread after finding the password
3. numThreads: The number of threads
4. encryptedPasswords: a string of encrypted passwords that need to be decrypted

**Methods:**

1. **initThreads()**
   1. Initializes each thread with the decryption function corresponding to each password to be decrypted
2. **startThreads()**
   1. Start all threads
3. **stopThreads()**
   1. Wait for all threads to complete
4. **ParallelDecryptionTime()**
   1. Measures the total time for parallel decryption

**3. SerialDecryptor Class**

This class implements serial decryption of all encrypted passwords, using a single instance of the Decryptor class.

**Properties:**

1. decryptor: instance of the decryptor class
2. encryptedPasswords: a string of encrypted passwords that need to be decrypted

**Methods:**

1. **SerialDecryption()**
   1. Run the bruteForceDecryption method for each password serially
2. **SerialDecryptionTime()**
   1. Measures the total time for serial decryption