

## Introducing the Caesar cipher

Tom Latham

## The Caesar Cipher

• Finally, we're ready to implement our first cipher

• A substitution cipher - each letter in the input text is replaced by another according to a constant rule

 Named after Julius Caesar - the first recorded user of this cipher!

# Caesar Cipher Encryption Substitution Rule

• Replace each letter in Plaintext string by that *K* letters **rightward** in the **Alphabet**.

• If the shift goes beyond the end of the **Alphabet**, wrap around to 'A' and continue counting **rightwards**.

• Shift K is an integer [0,25] and is the Key for the cipher

# Encrypting With the Caesar Cipher, K=5

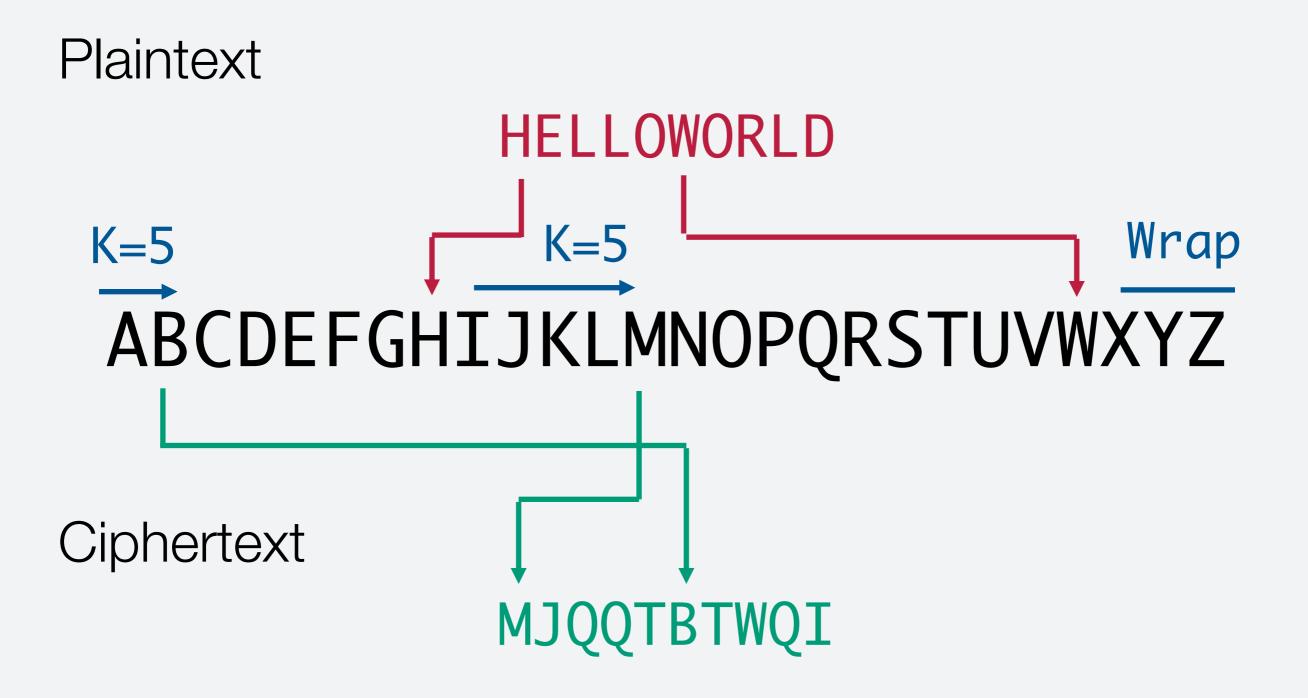
Plaintext

#### HELLOWORLD K=5 K=5 ABCDEFGHIJKLMNOPQRSTUVWXYZ

Ciphertext

MJQQT

# Encrypting With the Caesar Cipher, K=5



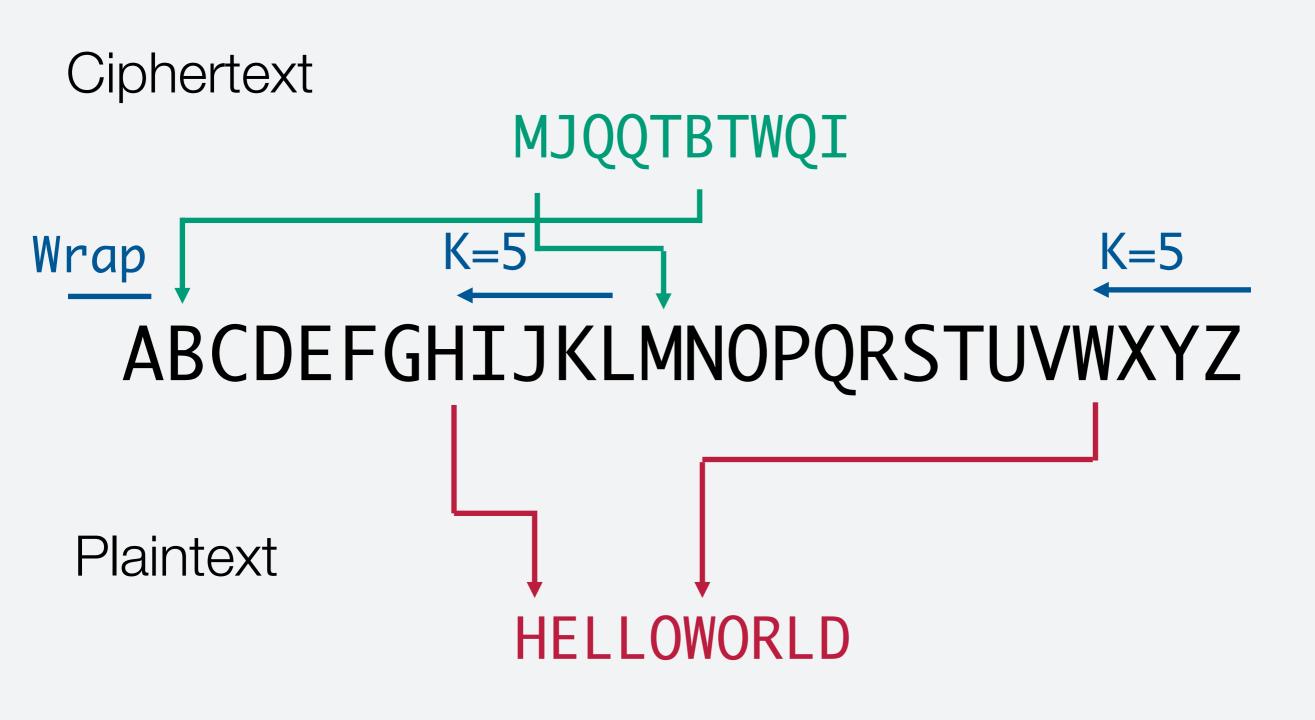
# Caesar Cipher Decryption Substitution Rule

 Replace each letter in CipherText by that K letters leftward in the Alphabet.

• If the shift goes beyond the start of the **Alphabet**, wrap around to 'Z' and continue counting **leftwards**.

• Shift K is an integer [0,25] and is the Key for the cipher

# **Decrypting With the Caesar Cipher, K=5**



## C++ Implementation

- Many ways to implement the Caesar Cipher in C++
- Today we're going to create a function called runCaesarCipher
- Let's think about what the interface of our function should be:
  - What inputs are needed?
  - What will the output be?
  - Hence what arguments should it have? And what return type?

• What else is involved?

## C++ Implementation

- Many ways to implement the Caesar Cipher in C++
- Today we're going to create a function called runCaesarCipher
- Let's think about what the interface of our function should be:
  - What inputs are needed? Input text, Key, Encrypt/Decrypt
  - What will the output be? Output text
  - Hence what arguments should it have? And what return type?

std::string runCaesarCipher( const std::string& inputText, const size\_t key, const bool encrypt )

Could have used a reference argument for the output text but since C++11 there is little efficiency gain and the intention is clearer this way.

• What else is involved? The alphabet

### C++ Implementation

// Create the alphabet container and output string

// Loop over the input text

}

// For each character find the corresponding position in the alphabet

// Apply the shift (+ve or -ve depending on encrypt/decrypt)
// to the position, handling correctly potential wrap-around

// Determine the new character and add it to the output string

// Finally (after the loop), return the output string

## **Exercise 5: implementing the Caesar Cipher**

- 1. Add handling of new command-line arguments that allow the user to:
  - a) Specify whether to encrypt or decrypt
  - b) Provide the cipher key
- 2. Implement the runCaesarCipher function (create new .hpp and .cpp files in the MPAGSCipher directory)
- 3. In your main function, use this function to encrypt/decrypt the transliterated text
- 4. You'll need to update the CMakeLists.txt file to build and link with this new code
- 5. When you have finished, commit and tag your repository (and push to github)
- ✓ There are some hints on the next slide to help with a few tricky points

## Implementation hints

- You will need to convert a string into an unsigned long to get the key from the command line – look at the online documentation: <u>http://en.cppreference.com/w/cpp/string/basic\_string</u>
- You can use either a std::vector<char> or a std::string to hold the alphabet
- To handle the "wrap-around", the modulus operator '%' could be useful
- Test that you have things working correctly by running the decrypt on your encrypted output
  - There are also online javascript implementations that you can check against