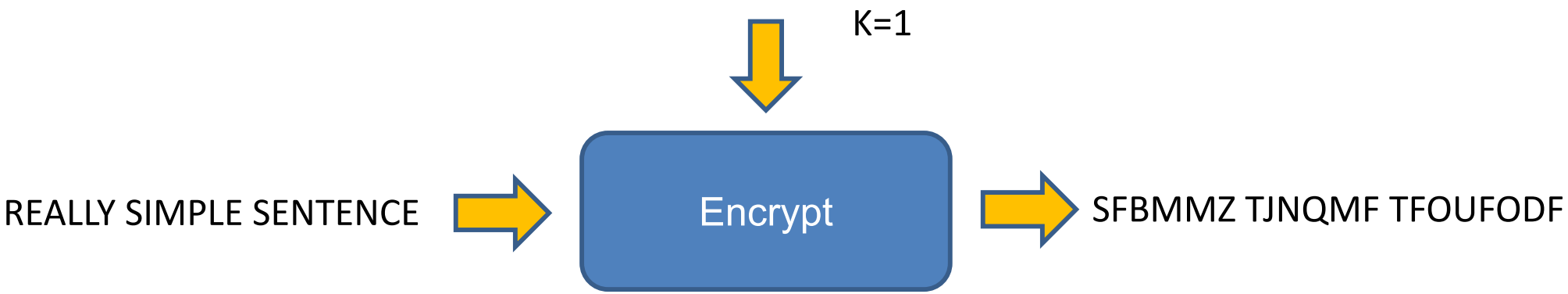
Caesar's encryption scheme

D. Ardagna, F. Filippini

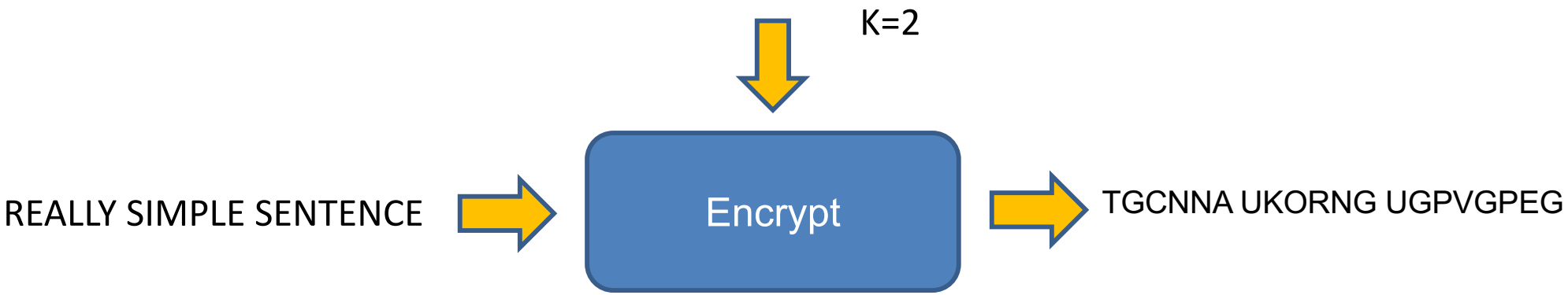
Symmetric Key Encryption

- Julius Caesar wanted to be sure that the messages sent to the chiefs of his legions could not be read by enemies
- He developed the following encryption algorithm:
  - Two people agree on a **key**, which is a **strictly positive number**
  - The sender adds the key to each letter, i.e., she/he replaces each letter with the one that goes key characters after it in the alphabet
  - The opposite procedure is followed by the receiver



Symmetric Key Encryption

- Julius Caesar wanted to be sure that the messages sent to the chiefs of his legions could not be read by enemies
- He developed the following encryption algorithm:
  - Two people agree on a **key**, which is a **strictly positive number**
  - The sender adds the key to each letter, i.e., she/he replaces each letter with the one that goes key characters after it in the alphabet
  - The opposite procedure is followed by the receiver



Symmetric Key Encryption

- Notice that
  - the **26 letters** of the English alphabet are considered in a circular way, so that **we start again with A when going after Z** (and vice versa)
  - **blank characters (e.g., spaces) are skipped** when performing the encryption. You can rely on the function `isblank`, that receives as input a character and returns **true** if it is blank

June 2021 Exam

1. implement a **serial function** with the following prototype:

```
std::string caesar (const std::string& str,
unsigned key, bool is_encrypted);
```

2. complete the main function, knowing that
   - **the key is provided by the user through the standard input**
   - the **sentence** is already provided but **known only to rank 0**
   - all ranks should call the serial function `caesar`, passing as input a portion of the initial sentence
   - assume that the length of the sentence (i.e., the total number of characters, spaces included) is a multiple of the number of available cores