Hw7:
the Happy Enigma
Part I

https://en.wikipedia.org/wiki/Enigma_machine
https://en.wikipedia.org/wiki/Enigma_rotor_details
http://www.enigmaco.de/enigma/ <- DEMO
Due date: 6/10 (SUN.)

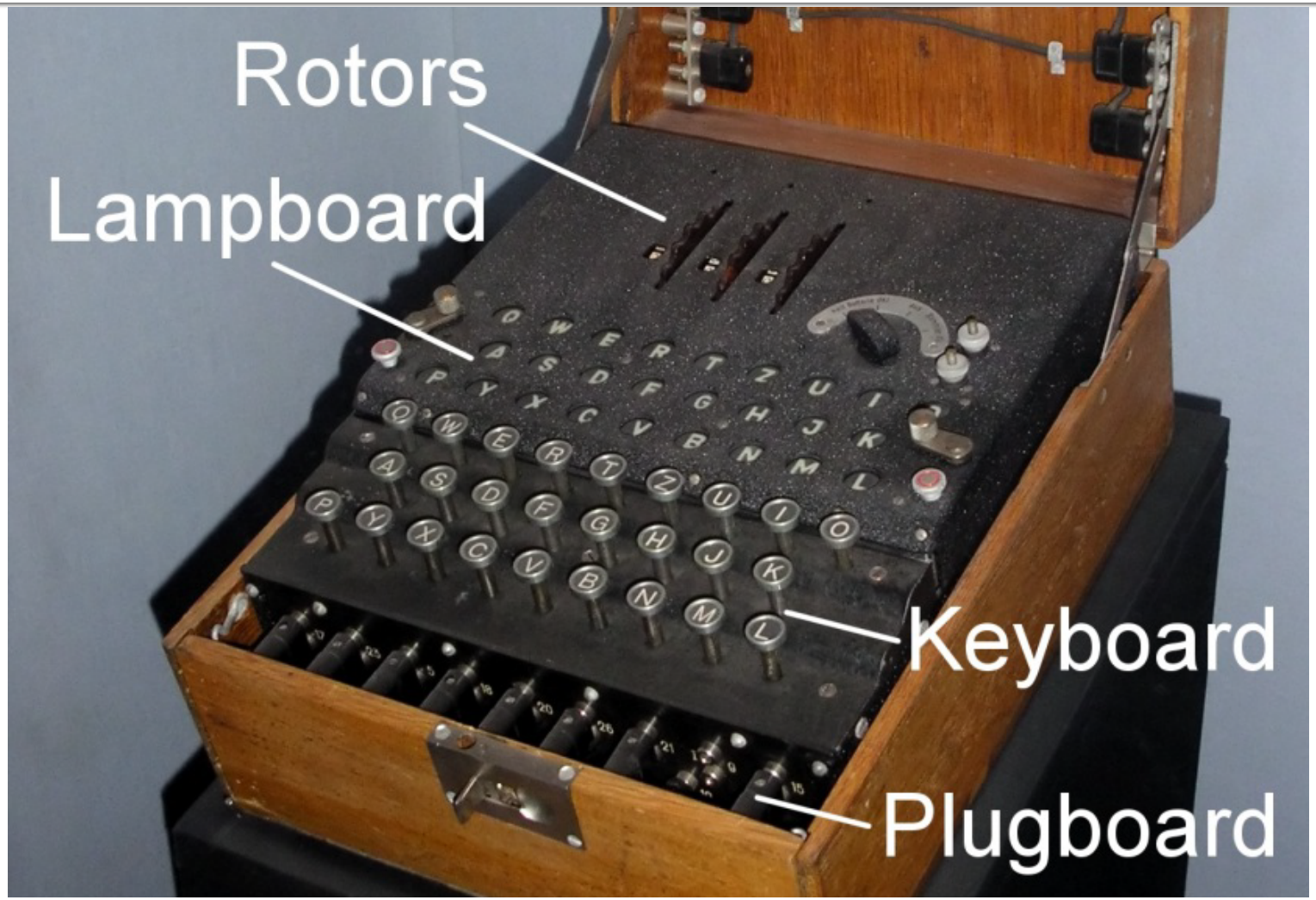
Introduction of Enigma

- 在密碼學史中，恩尼格瑪密碼機（德語：Enigma，又譯啞謎機，或謎）是一種用於加密與解密文件的密碼機。確切地說，Enigma是對二戰時期納粹德國使用的一系列相似的旋轉機加解密機器的統稱，它包括了許多不同的型號。

- Enigma密碼機在1920年代早期開始被用於商業，一些國家的軍隊與政府也曾使用過它，其中的主要使用者是第二次世界大戰時的納粹德國。

http://zh.wikipedia.org/wiki/%E6%81%A9%E5%B0%BC%E6%A0%BC%E7%8E%9B%E5%AF%86%E7%A0%81%E6%9C%BA
https://en.wikipedia.org/wiki/Enigma_machine

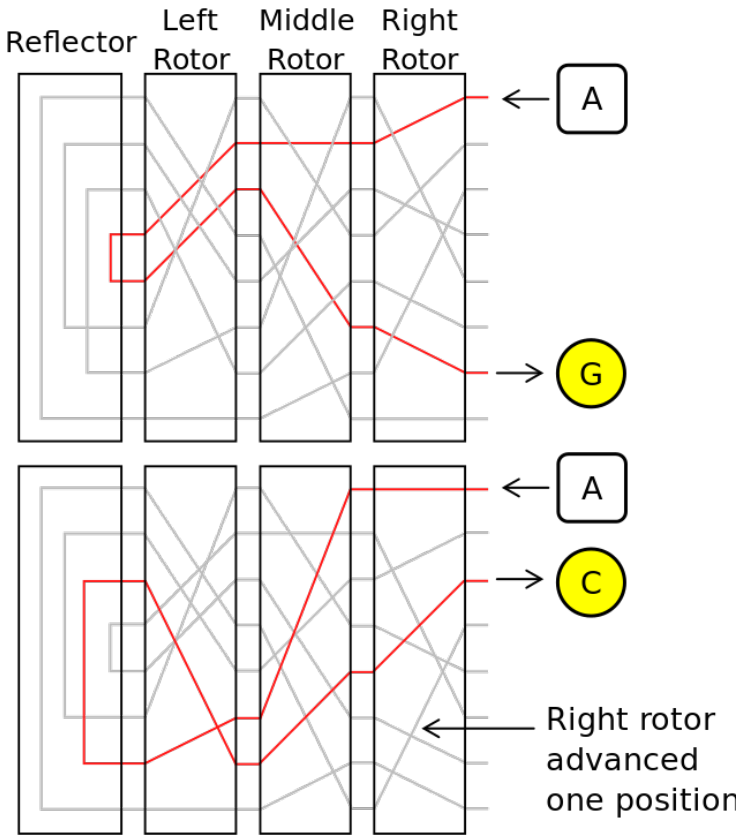
Introduction of Enigma



https://en.wikipedia.org/wiki/Enigma_machine#/media/File:EnigmaMachineLabeled.jpg

Enigma

ENIGMA旋轉盤的工作原理圖

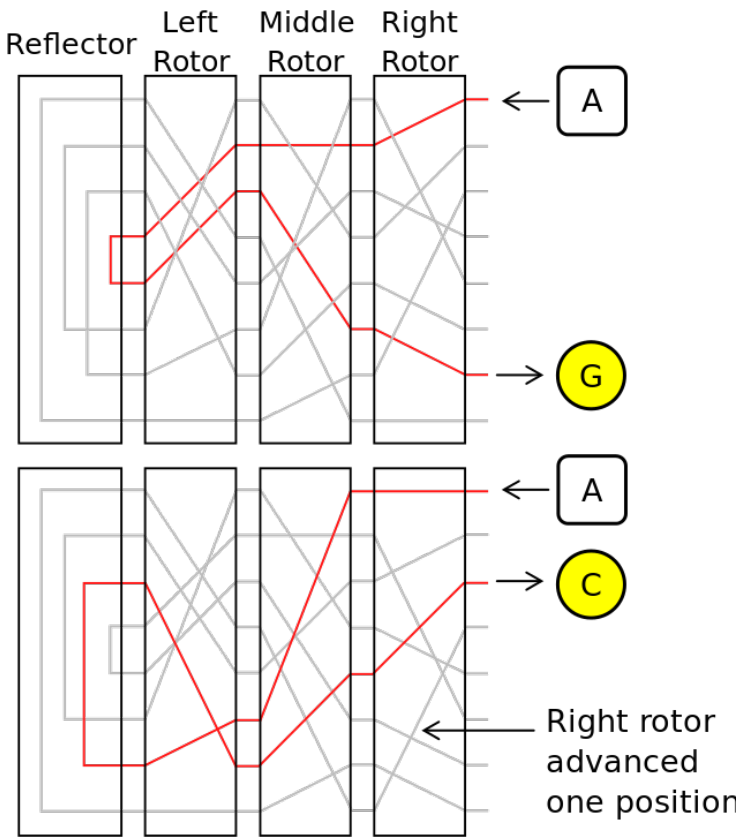


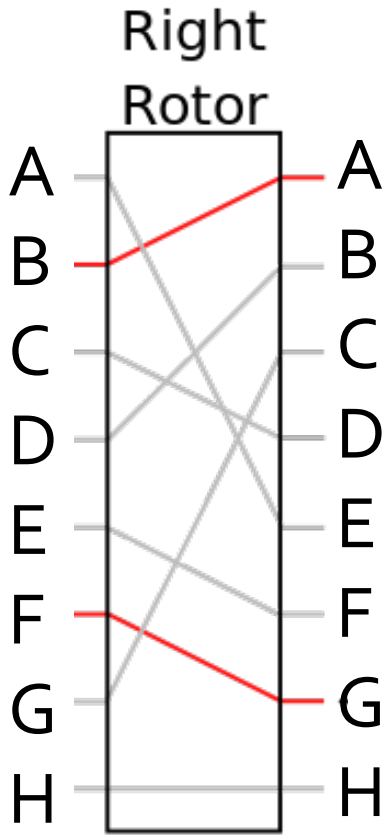
- 連續按兩次A鍵後，電流會流經所有旋轉盤 (Rotor)，通過反射器 (Reflector)後分別向反方向流到G燈和C燈。
- 連續按兩次A鍵會得到不同的結果，這是因為Right Rotor在第一次按下A鍵後會旋轉，這就將A鍵發出的電流送到了一個完全不同的路線上。

http://zh.wikipedia.org/wiki/%E6%81%A9%E5%B0%BC%E6%A0%BC%E7%8E%9B%E5%AF%86%E7%A0%81%E6%9C%BA

Enigma

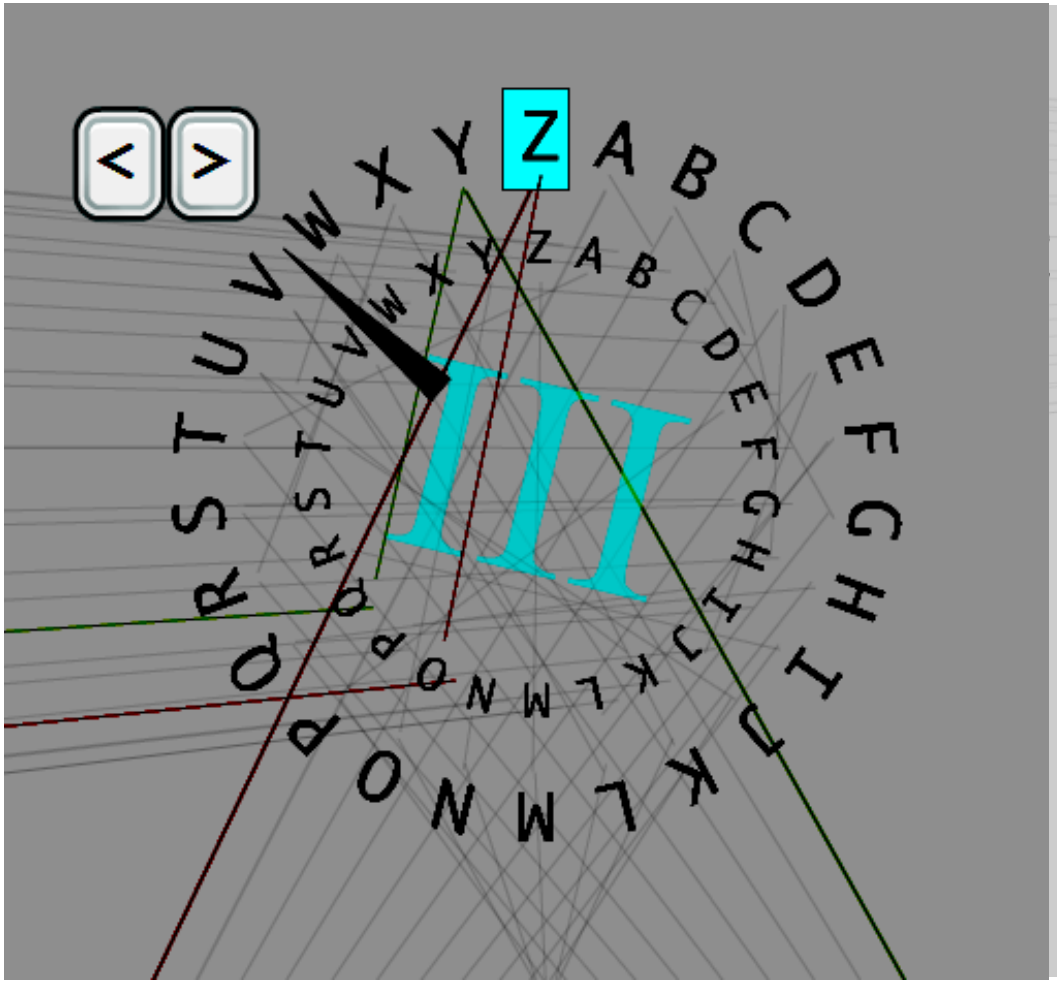
ENIGMA旋轉盤的工作原理圖





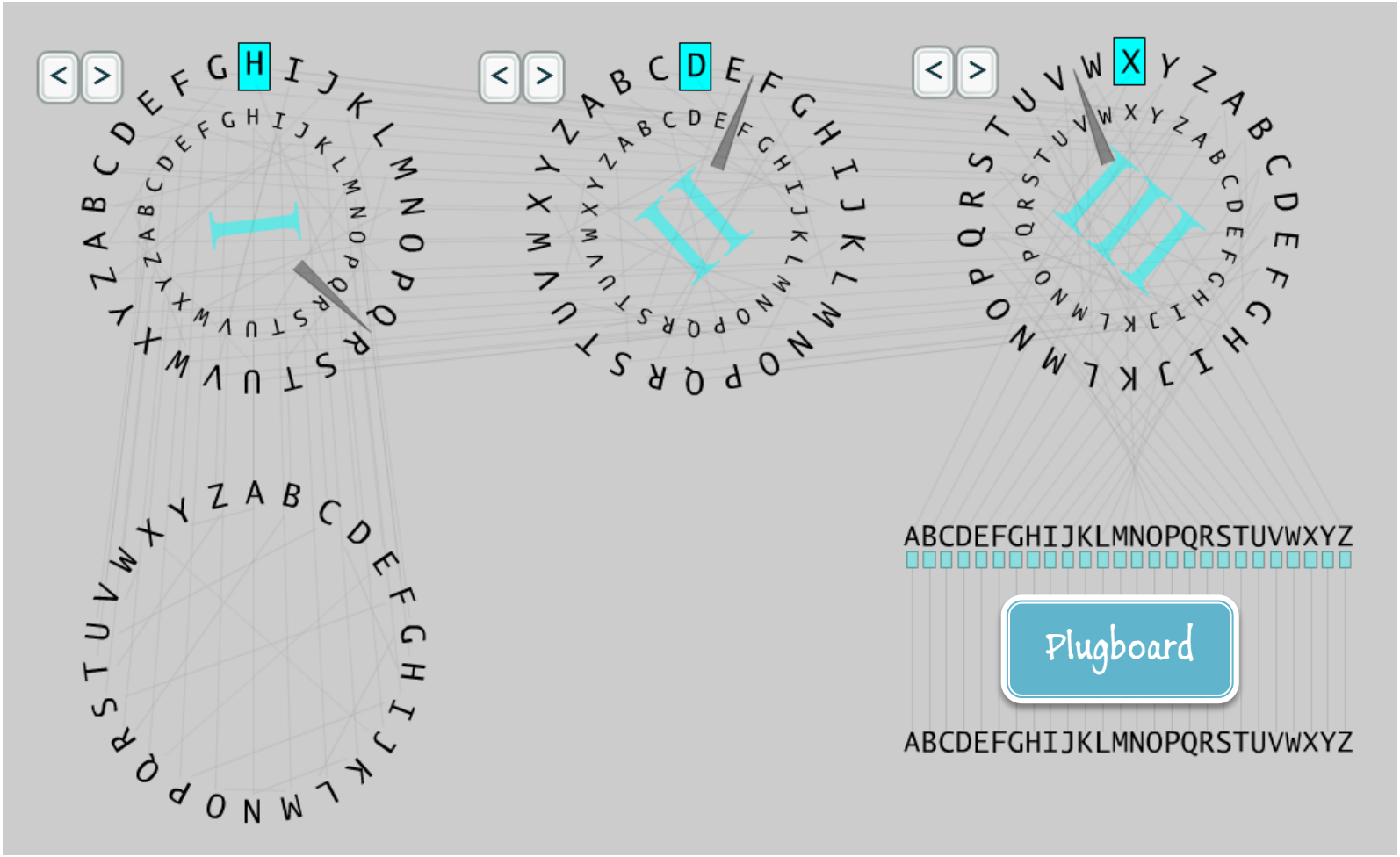
http://zh.wikipedia.org/wiki/%E6%81%A9%E5%B0%BC%E6%A0%BC%E7%8E%9B%E5%AF%86%E7%A0%81%E6%9C%BA

Enigma



http://www.enigmaco.de/enigma/ ←主要請參考這個

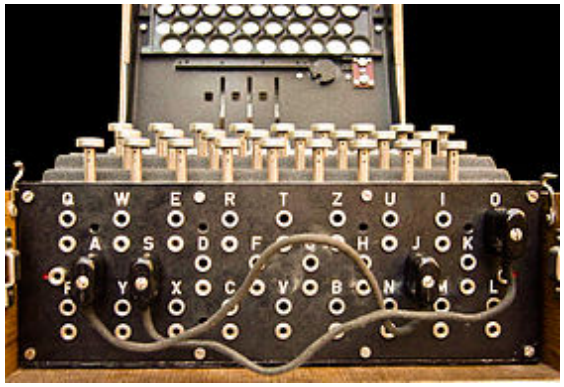
Enigma



http://www.enigmaco.de/enigma/

Enigma

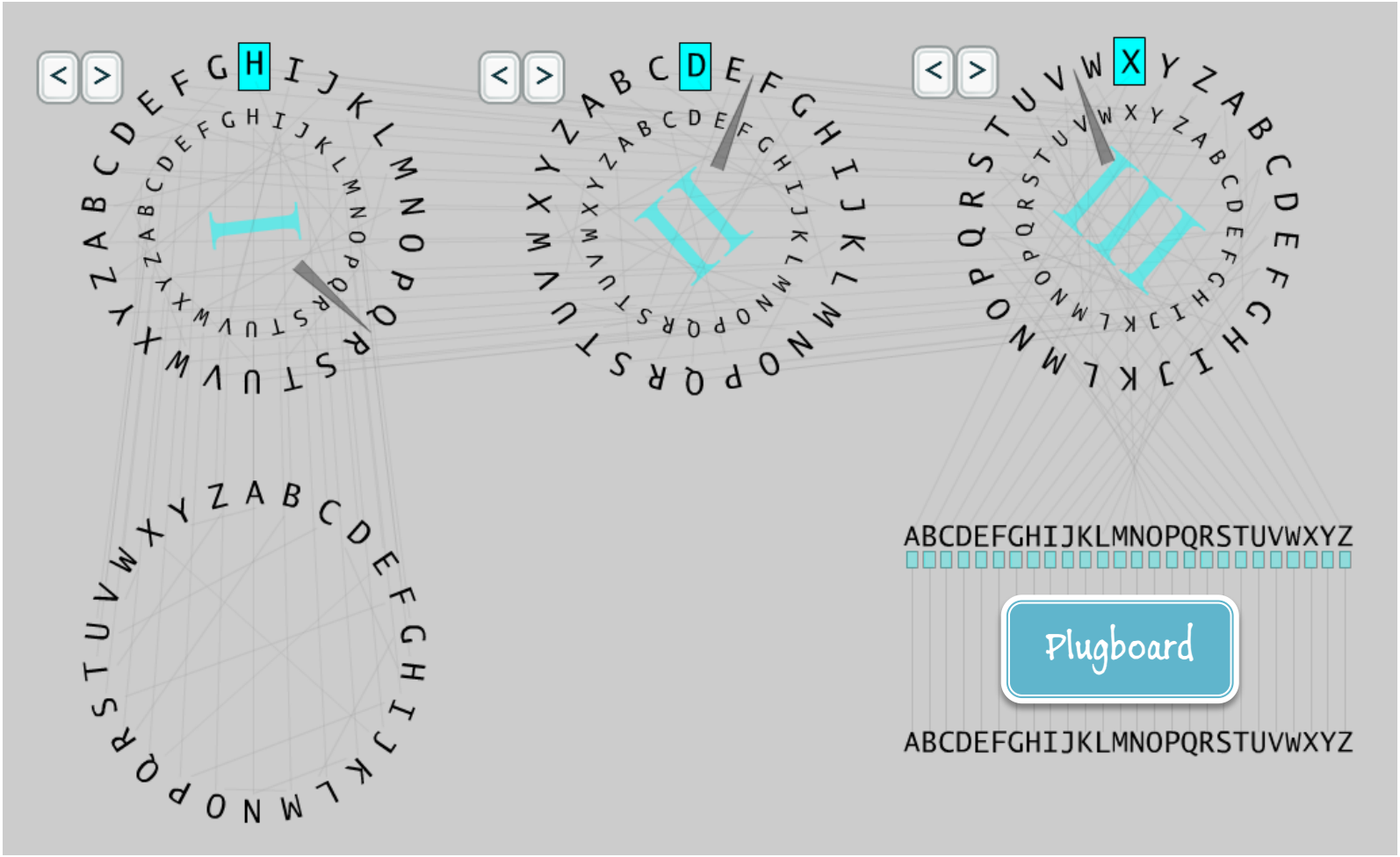
Plugboard (接線板)
接線板位於Enigma前部鍵盤的下方，每條線都會連接一對字母。
這些線的作用就是在電流進入旋轉盤前改變它的方向。
而接線板上最多可以同時接13條線。



右圖中
接線板上共有兩對字母被連接起來
S-O
J-A

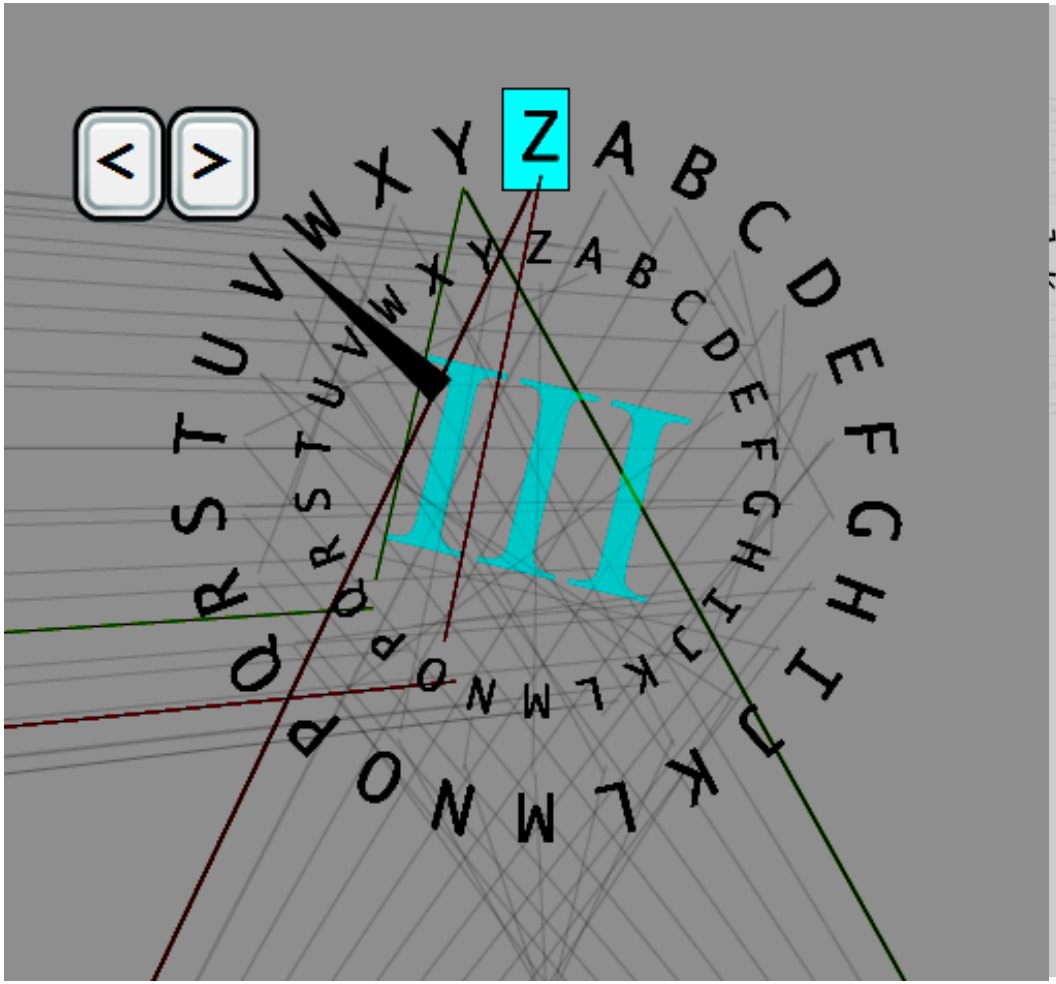
http://www.enigmaco.de/enigma/

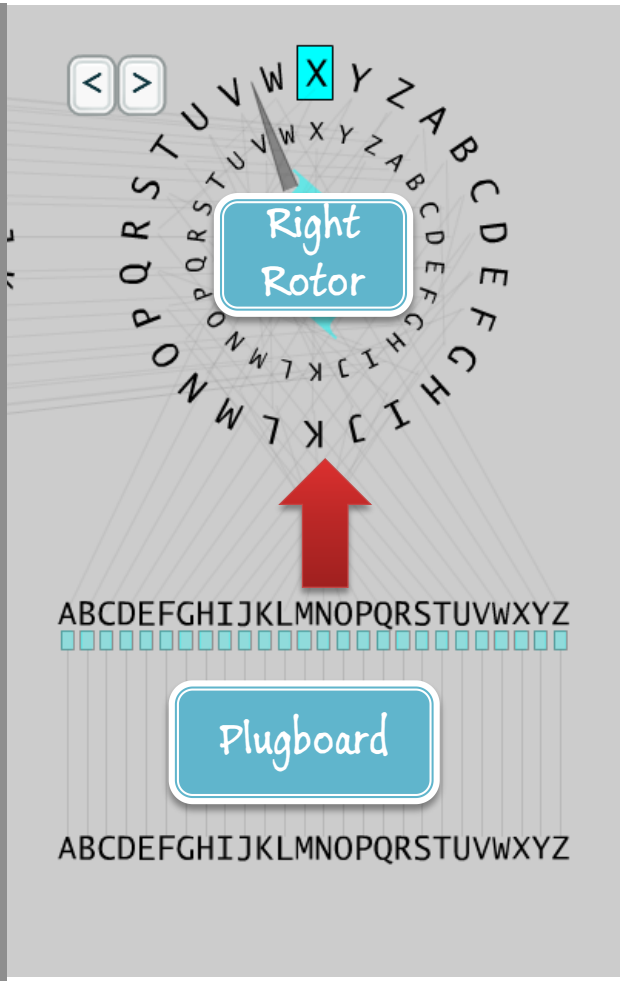
Enigma



http://www.enigmaco.de/enigma/

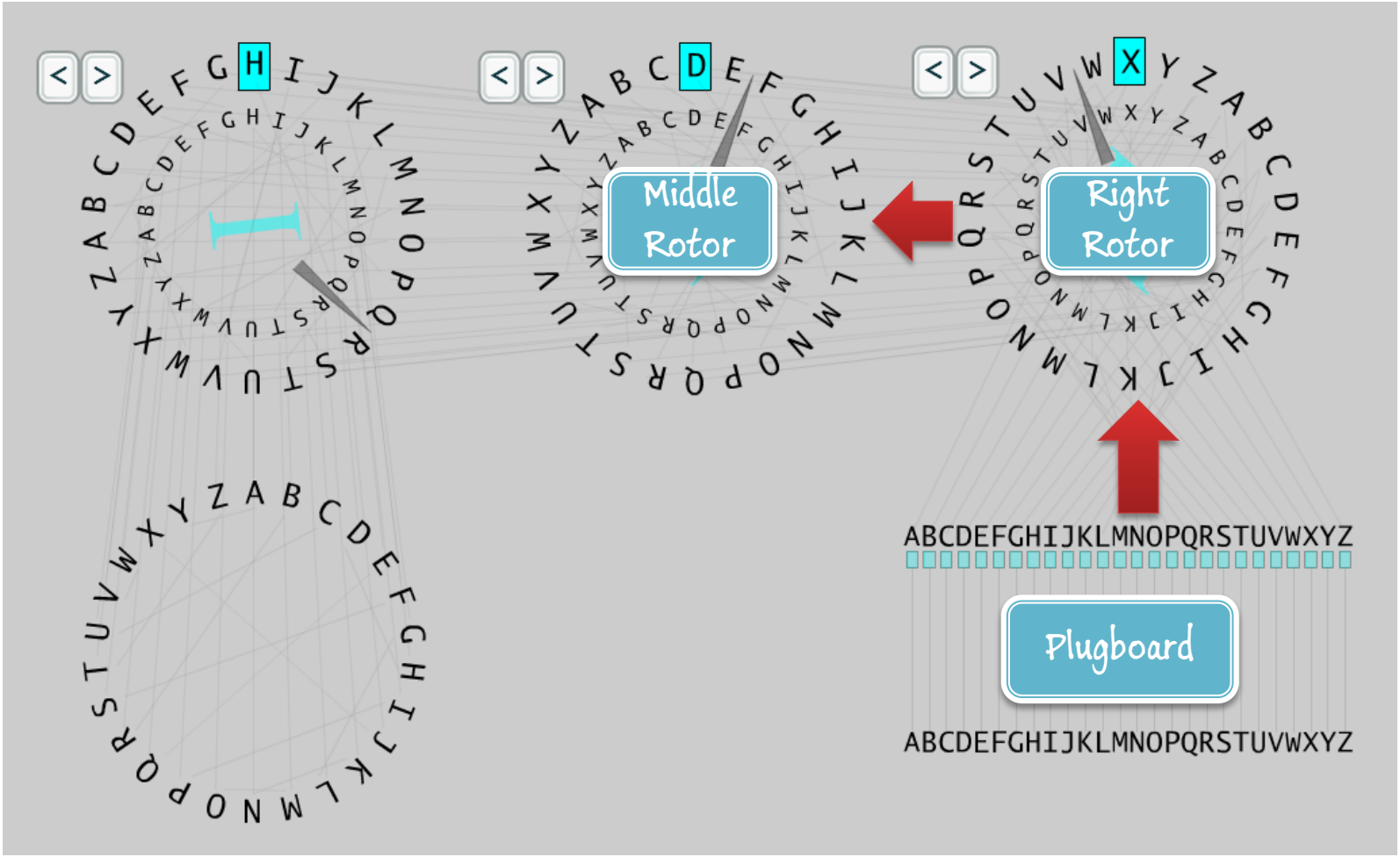
Enigma





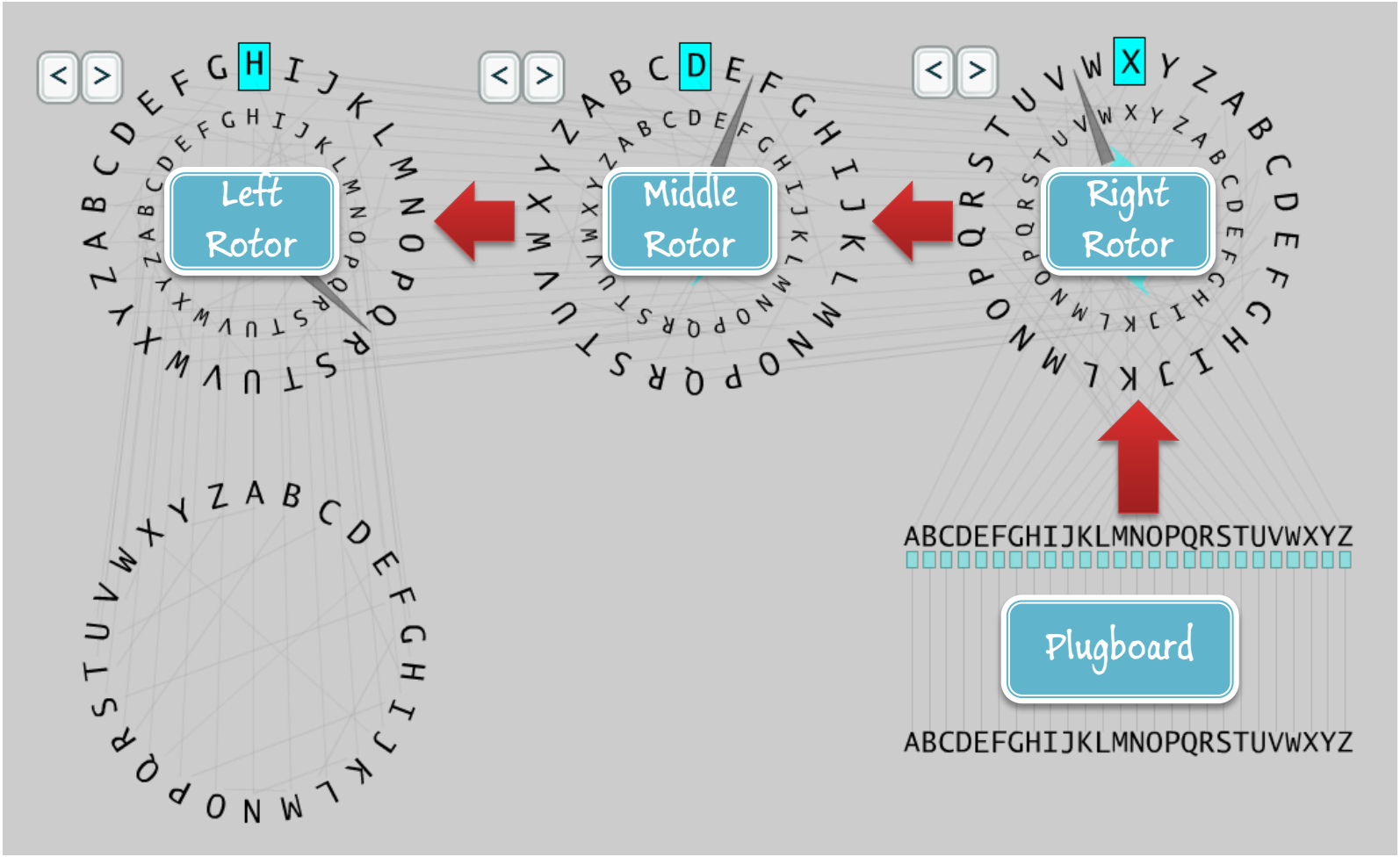
http://www.enigmaco.de/enigma/

Enigma



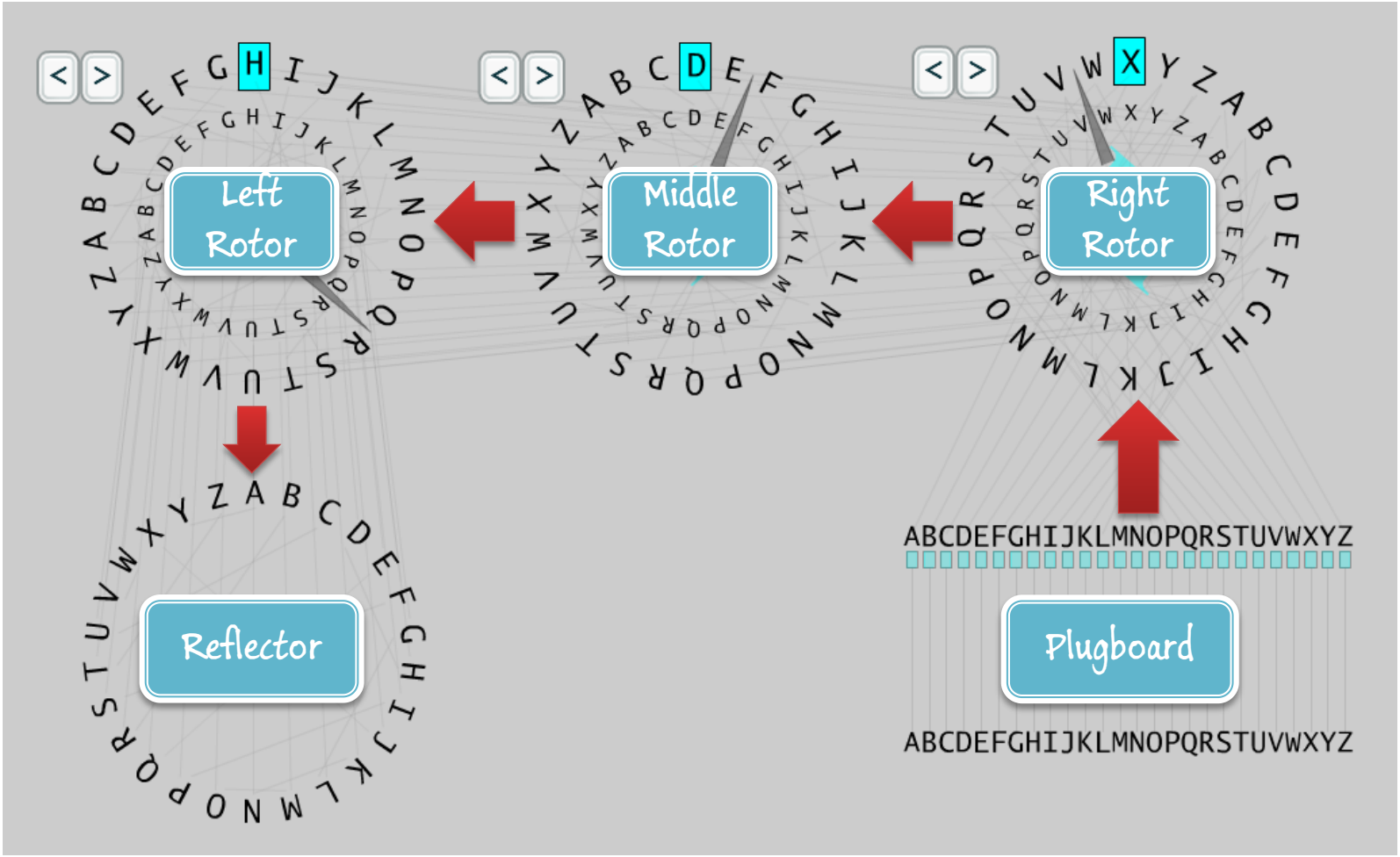
http://www.enigmaco.de/enigma/

Enigma



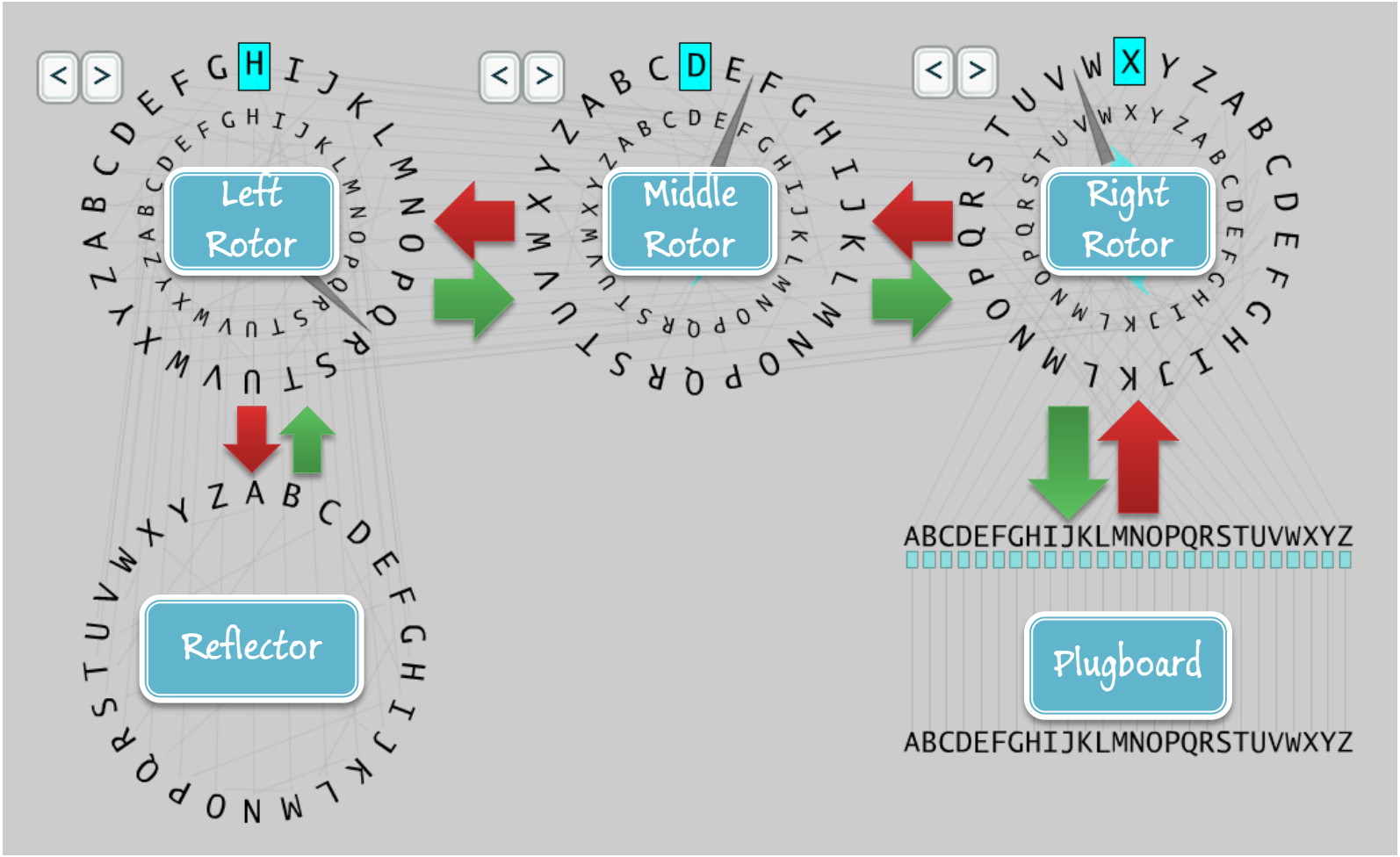
http://www.enigmaco.de/enigma/

Enigma



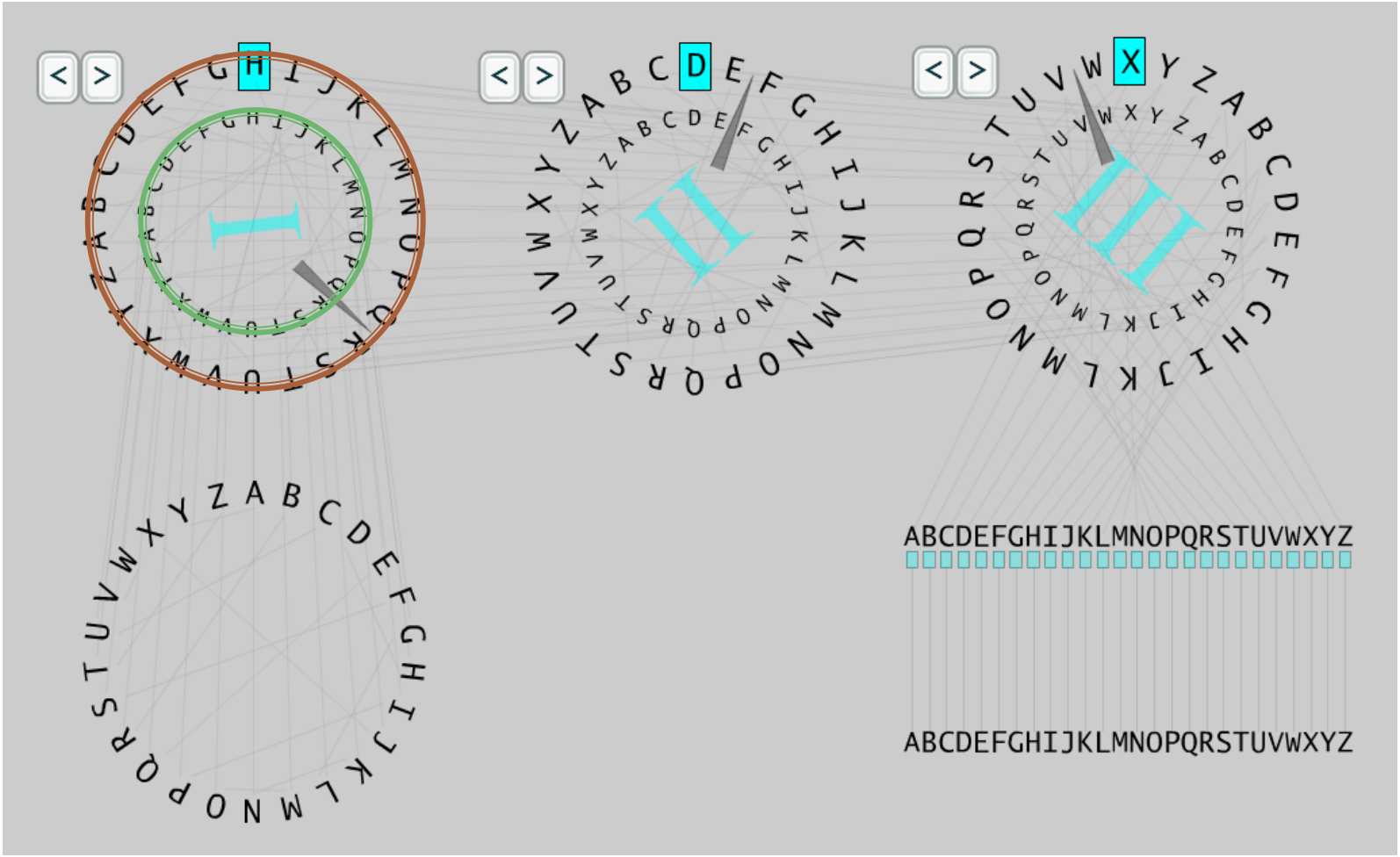
http://www.enigmaco.de/enigma/

Enigma



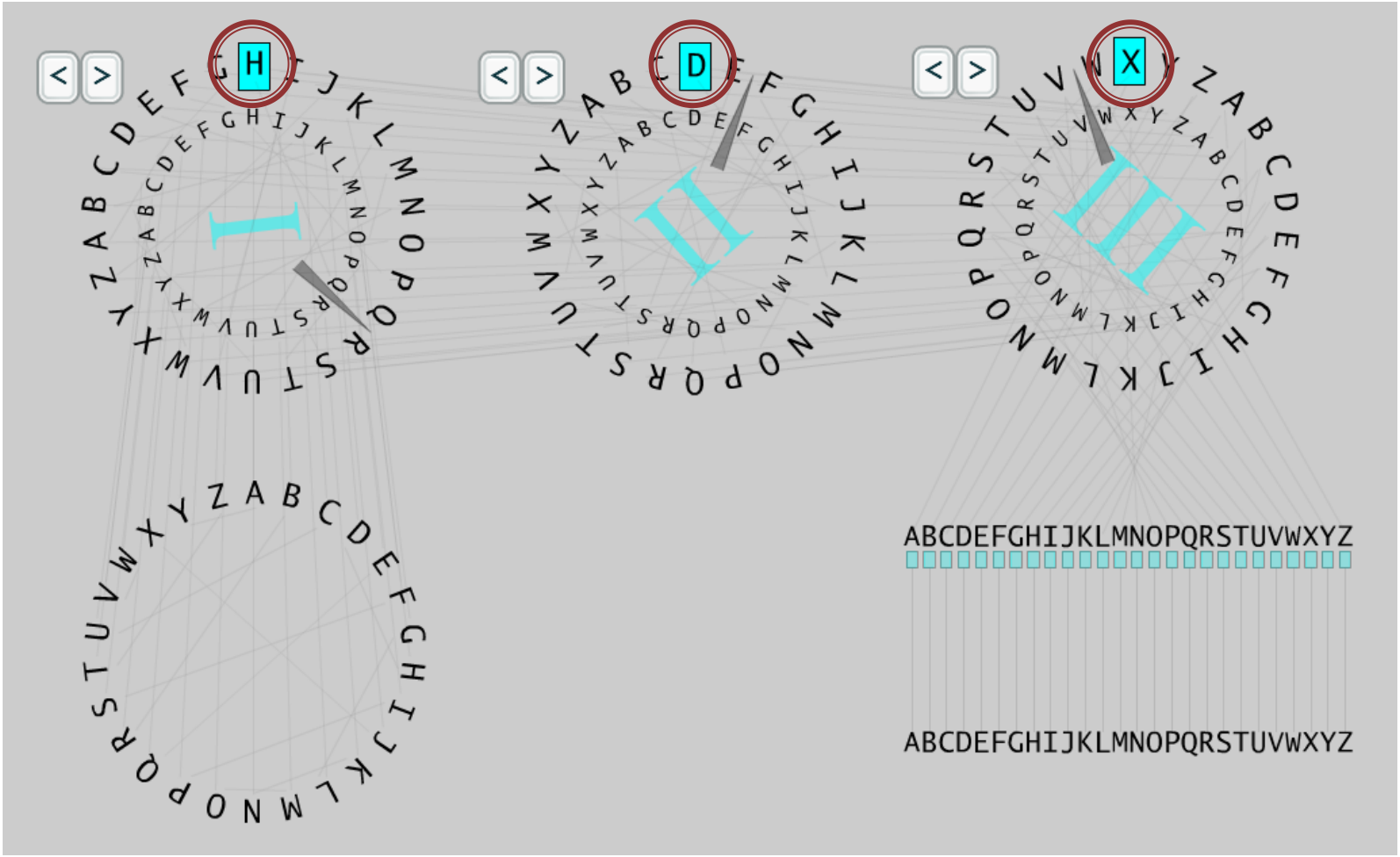
http://www.enigmaco.de/enigma/

Rotors of Enigma



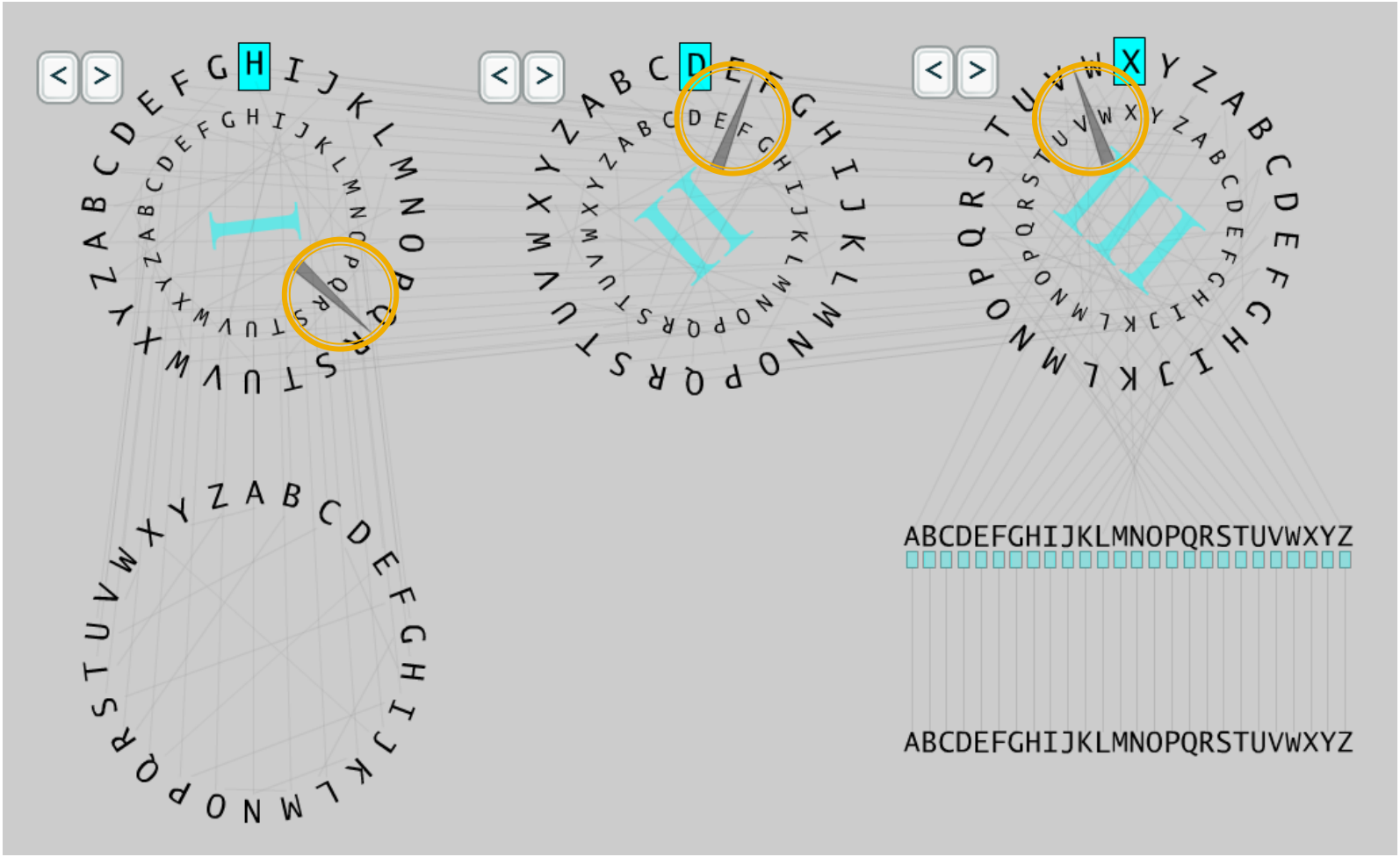
http://www.enigmaco.de/enigma/

Rotors of Enigma



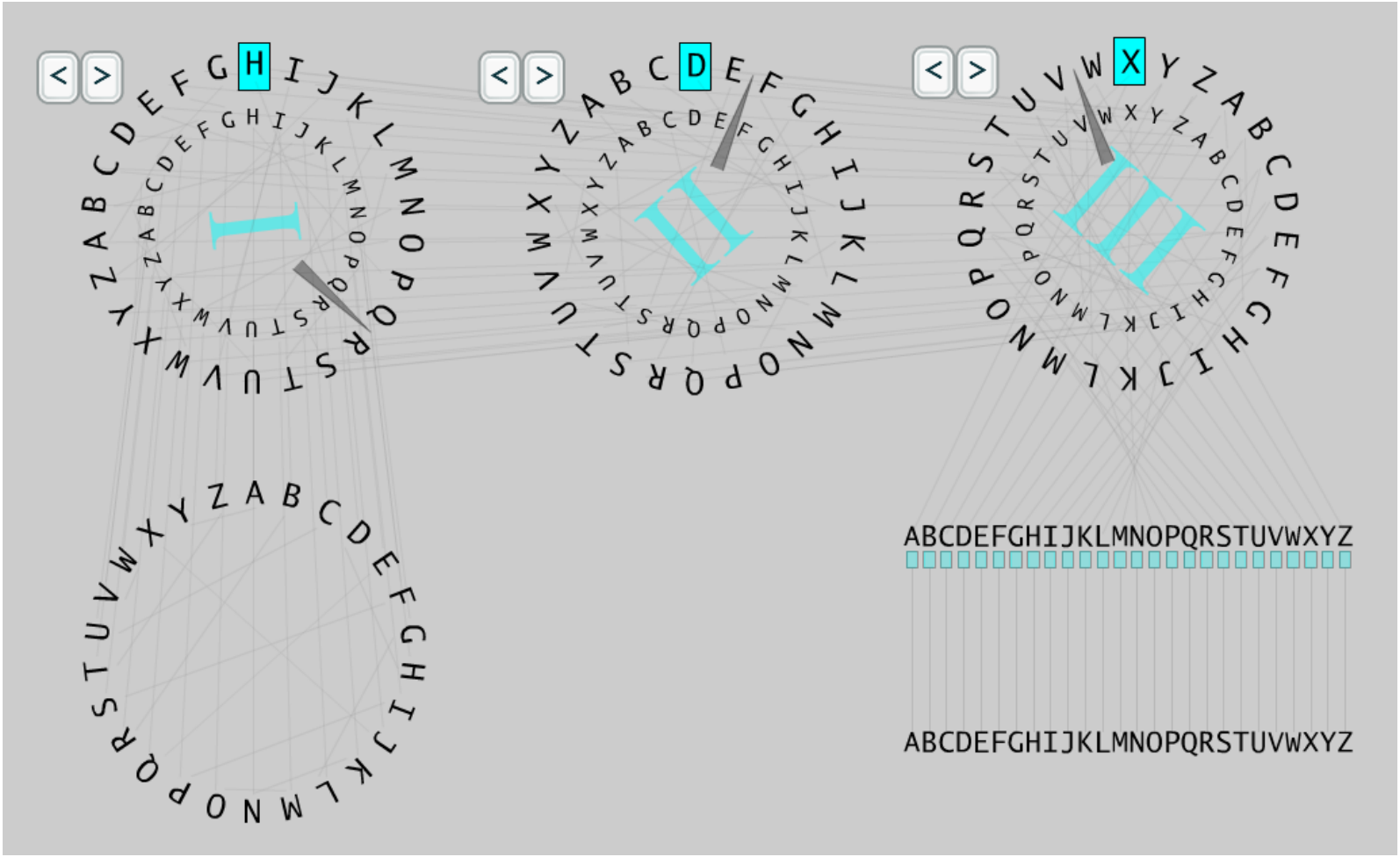
http://www.enigmaco.de/enigma/

Rotors of Enigma



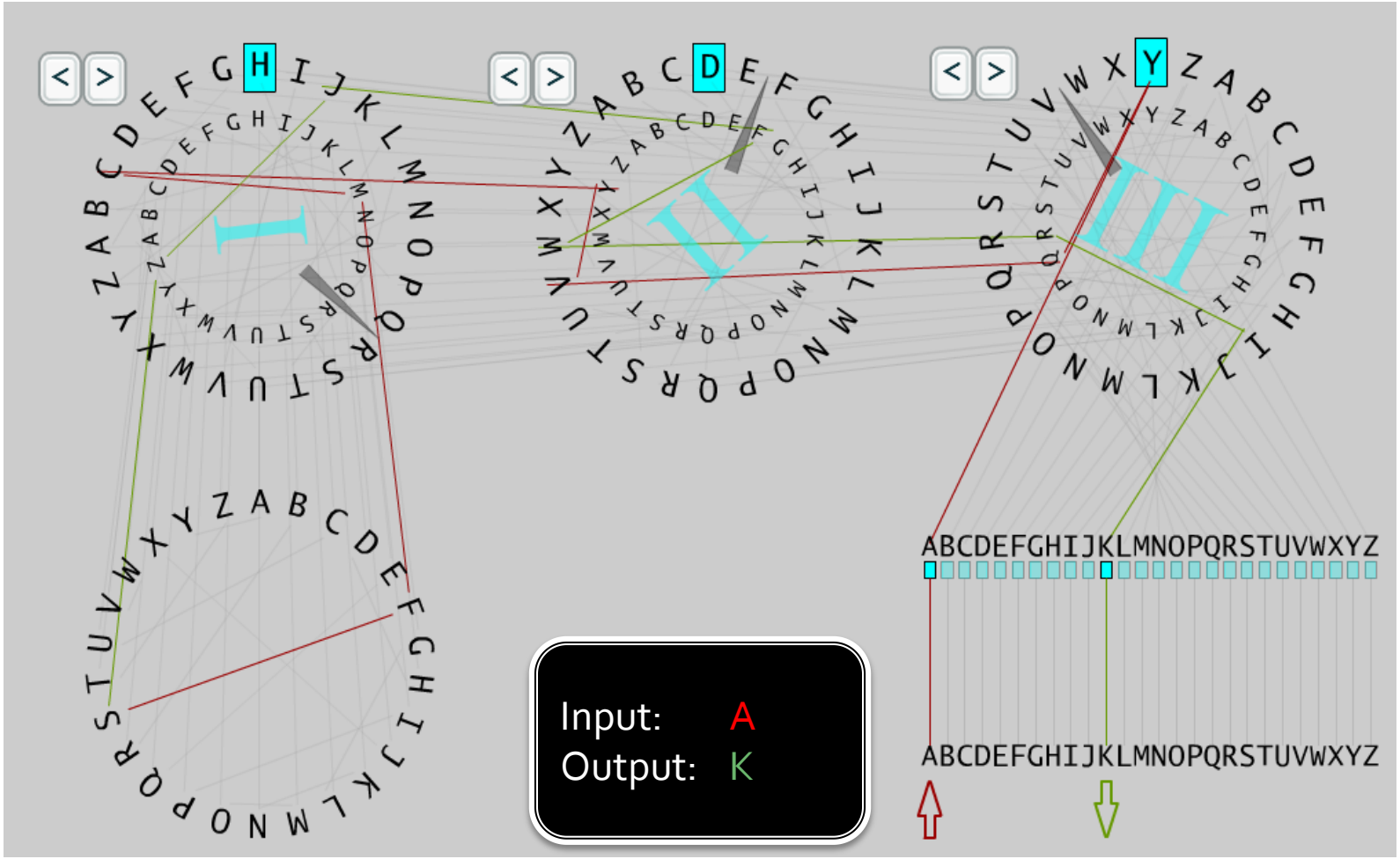
http://www.enigmaco.de/enigma/

Rotors of Enigma



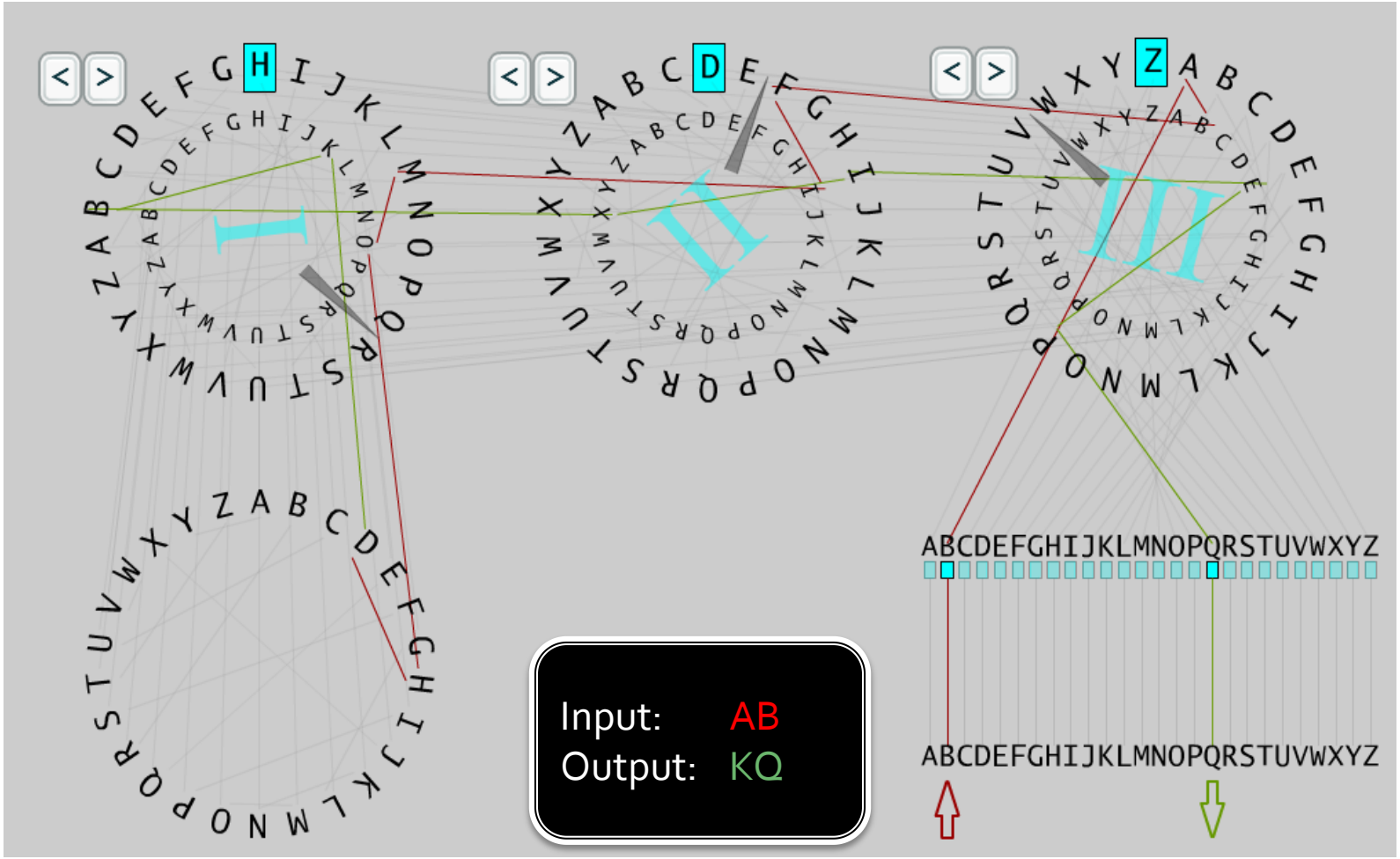
http://www.enigmaco.de/enigma/

Enigma



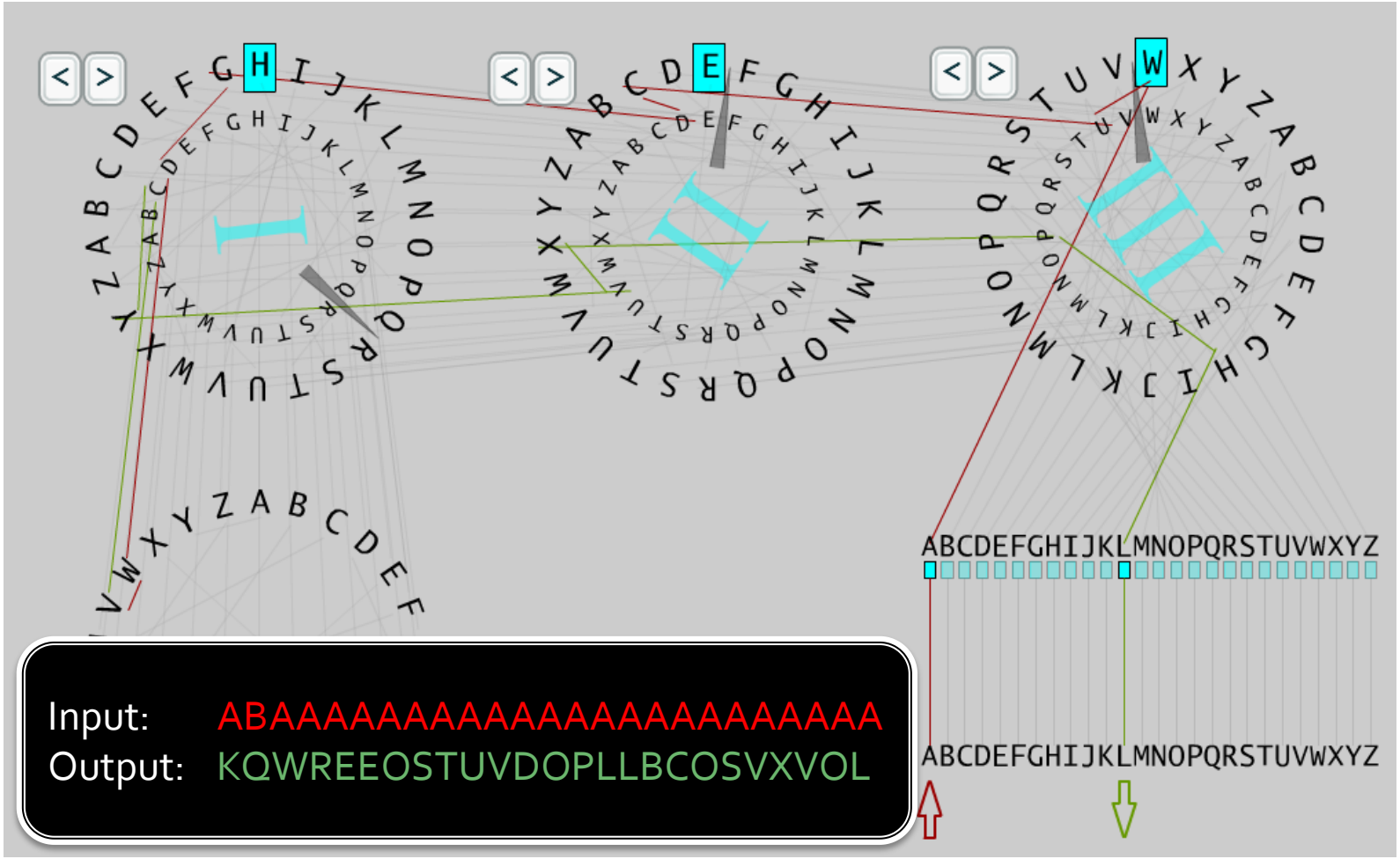
http://www.enigmaco.de/enigma/

Enigma



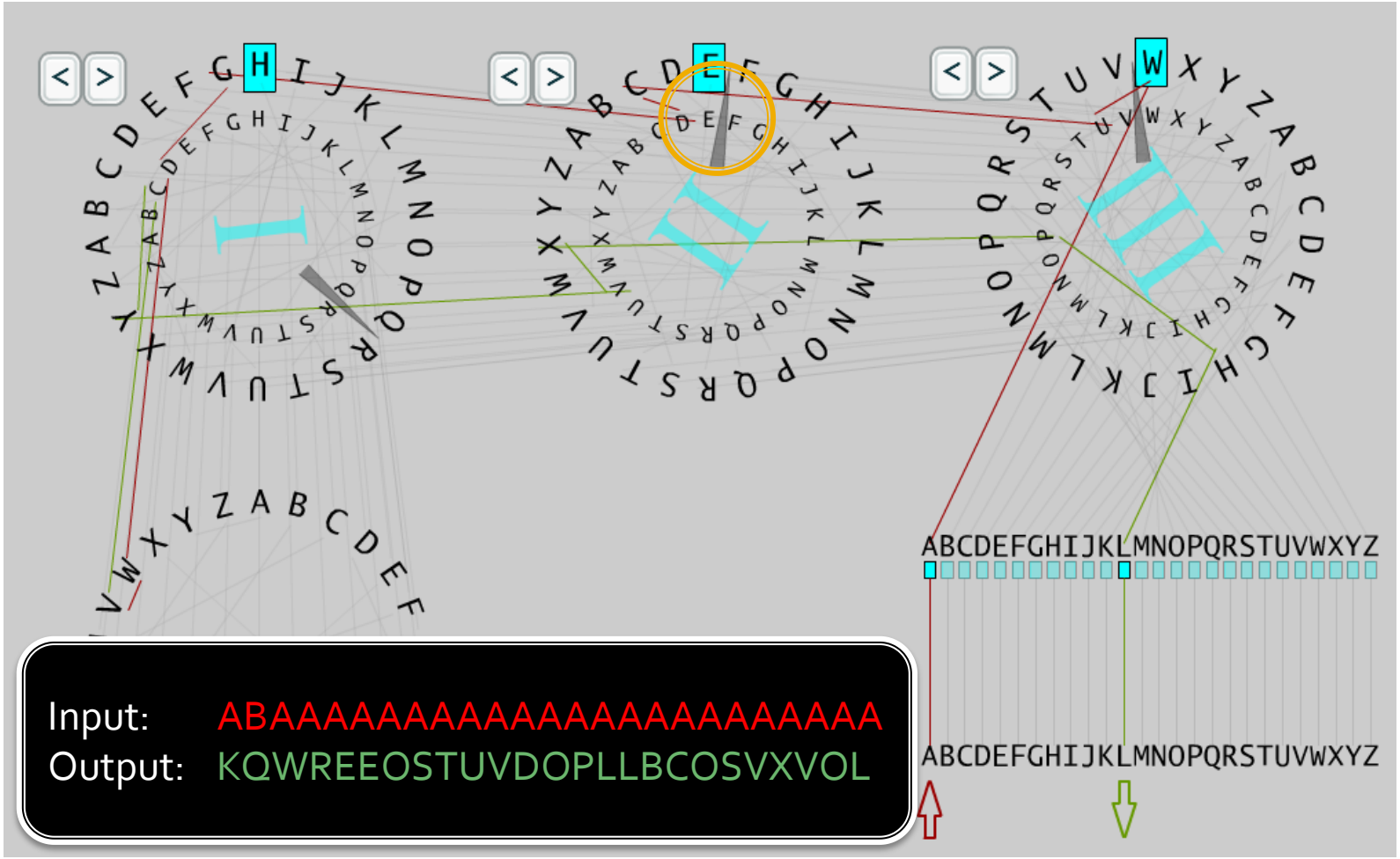
http://www.enigmaco.de/enigma/

Enigma



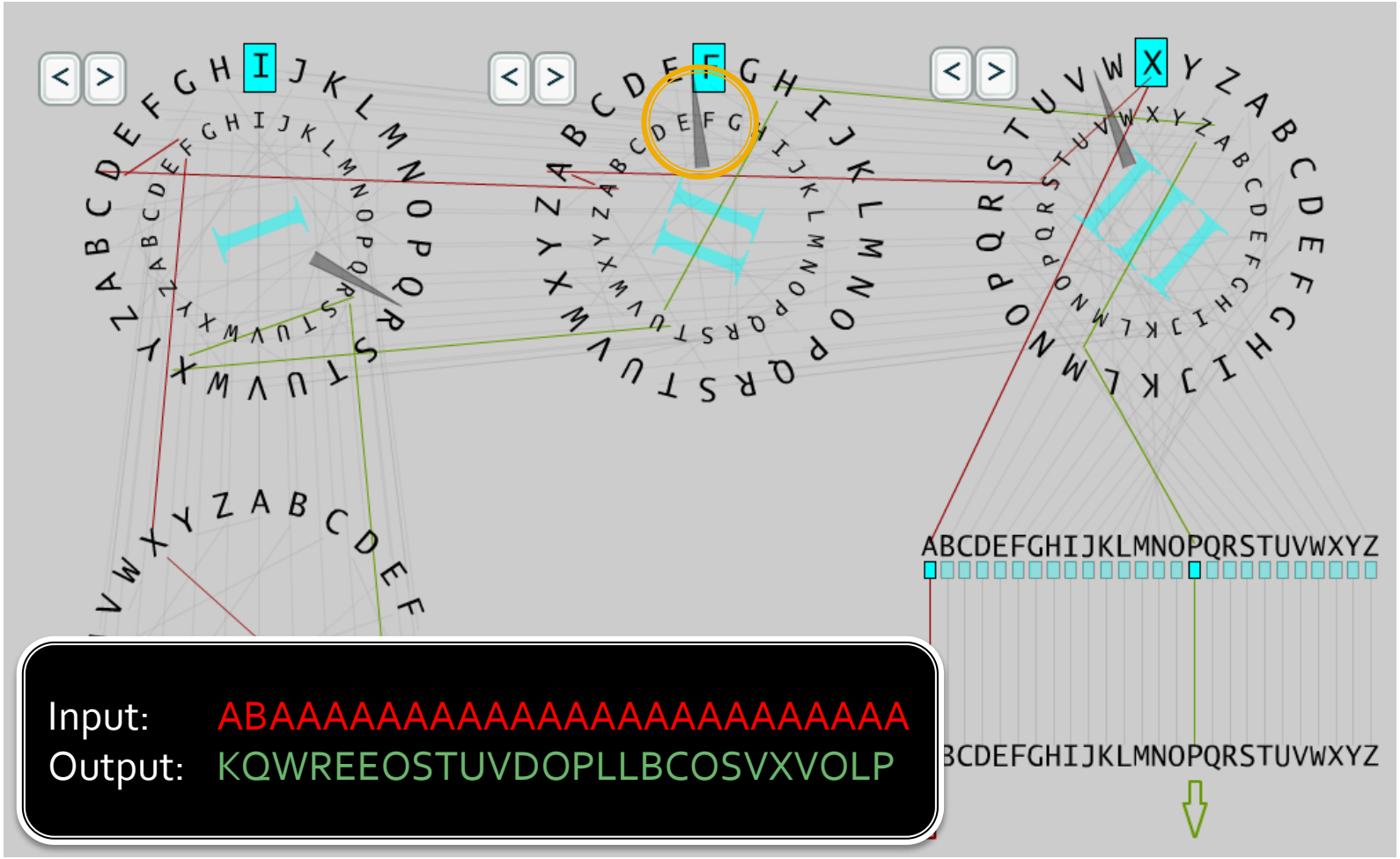
http://www.enigmaco.de/enigma/

Special Rule of Enigma Rotor II



http://www.enigmaco.de/enigma/

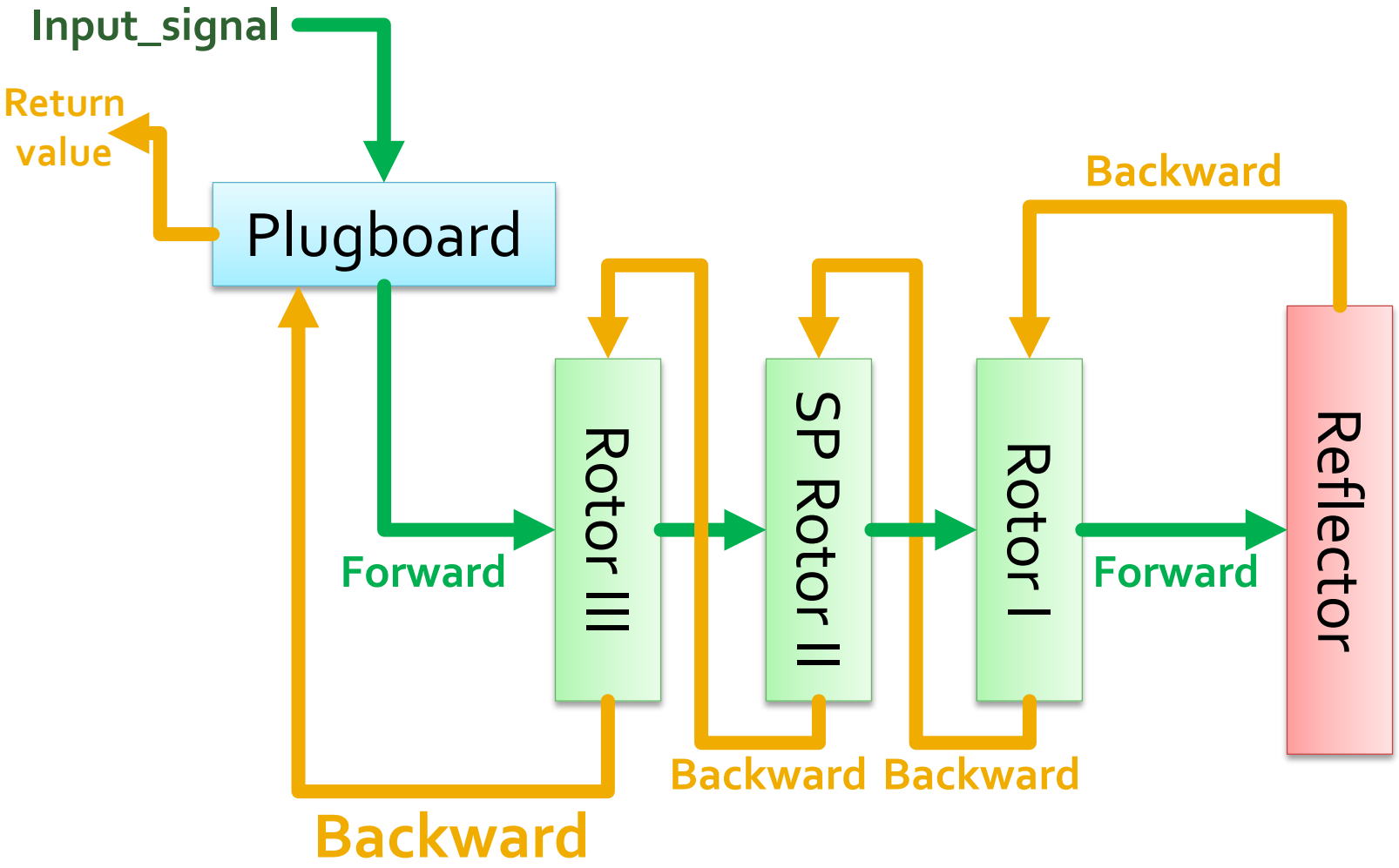
Special Rule of Enigma Rotor II



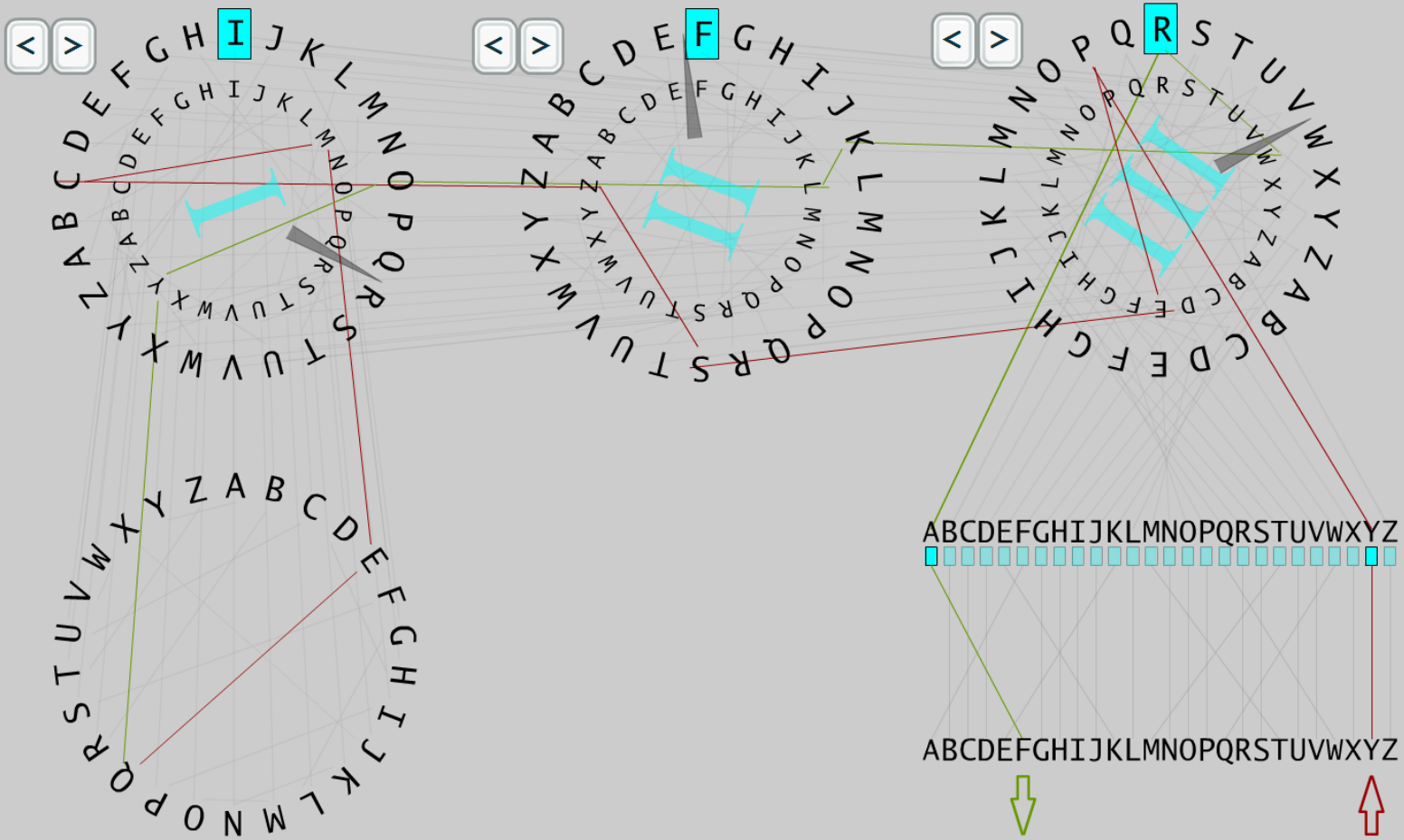
http://www.enigmaco.de/enigma/

the Happy Enigma
Part I

Original Structure



***You may use the RETURN value as the backward path.



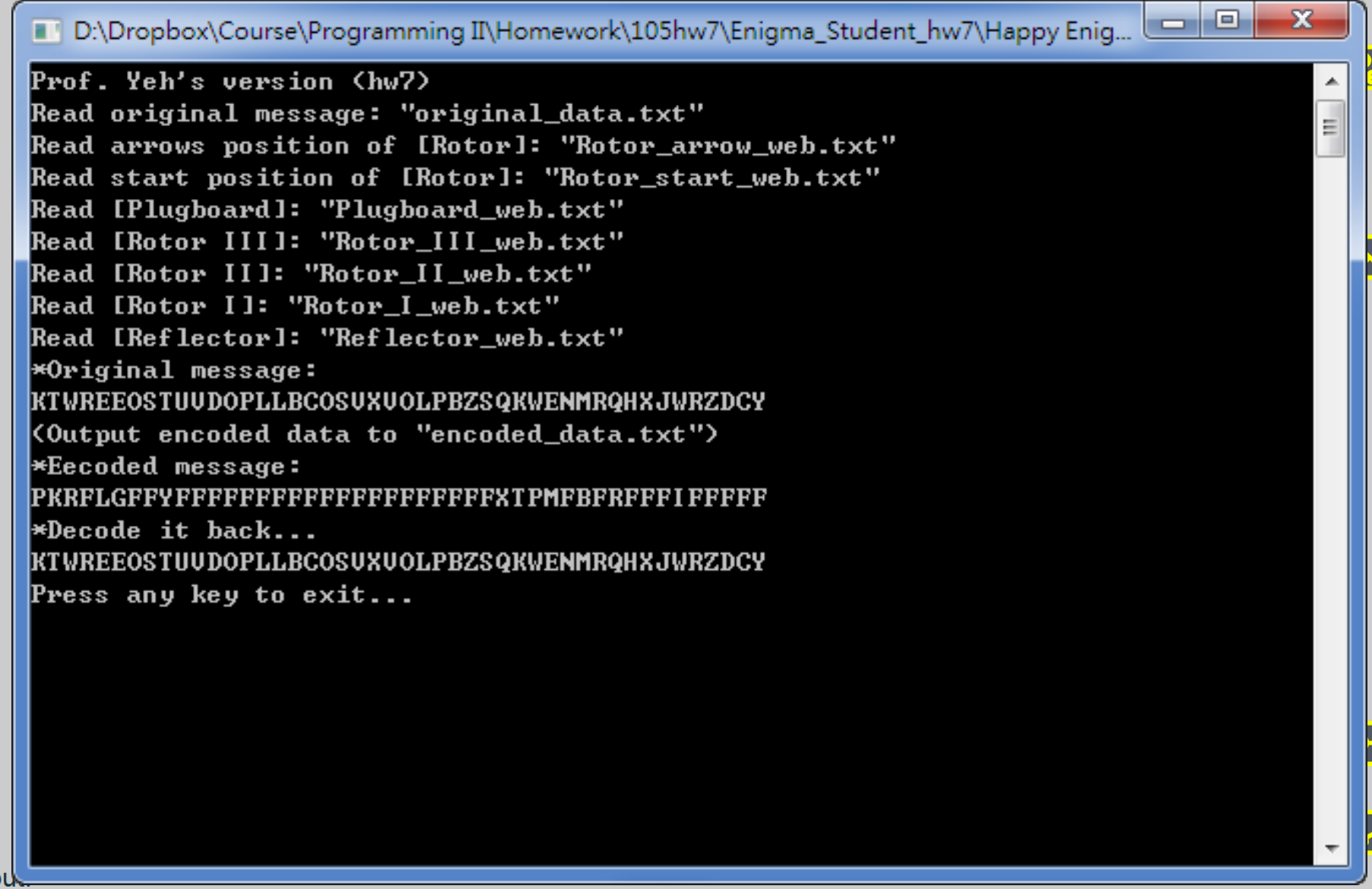
Input:

KTWREEOSTUVDOPLLBCOSVXVOLPBZSQKWENMRQHXJWRZDCY

Output:

PKRFLGFFYFFFFFFFFFFFFFFFFFFFXTPMFBFRFFFIFFFFF

Status: Highlighted wires show steps of encryption.



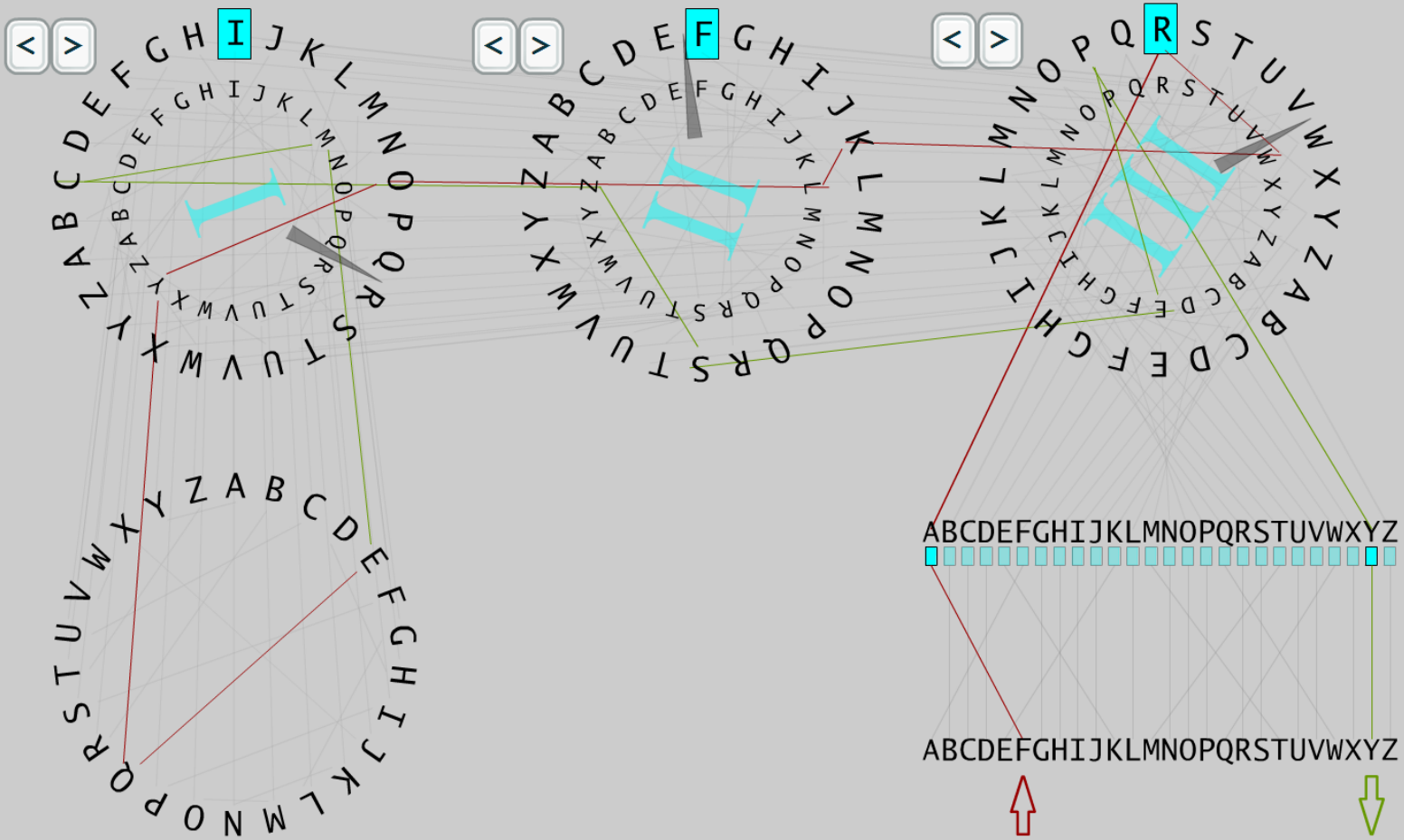
Input:

KTWREEOSTUVDOPLLBCOSVXVOLPBZSQKWENMRQHXJWRZDCY

Output:

PKRFLGFFYFFFFFFFFFFFFFFFFFFFXTPMFBFRFFFIFFFFF

Status: Highlighted wires show steps of encryption.



Reverse

Input:

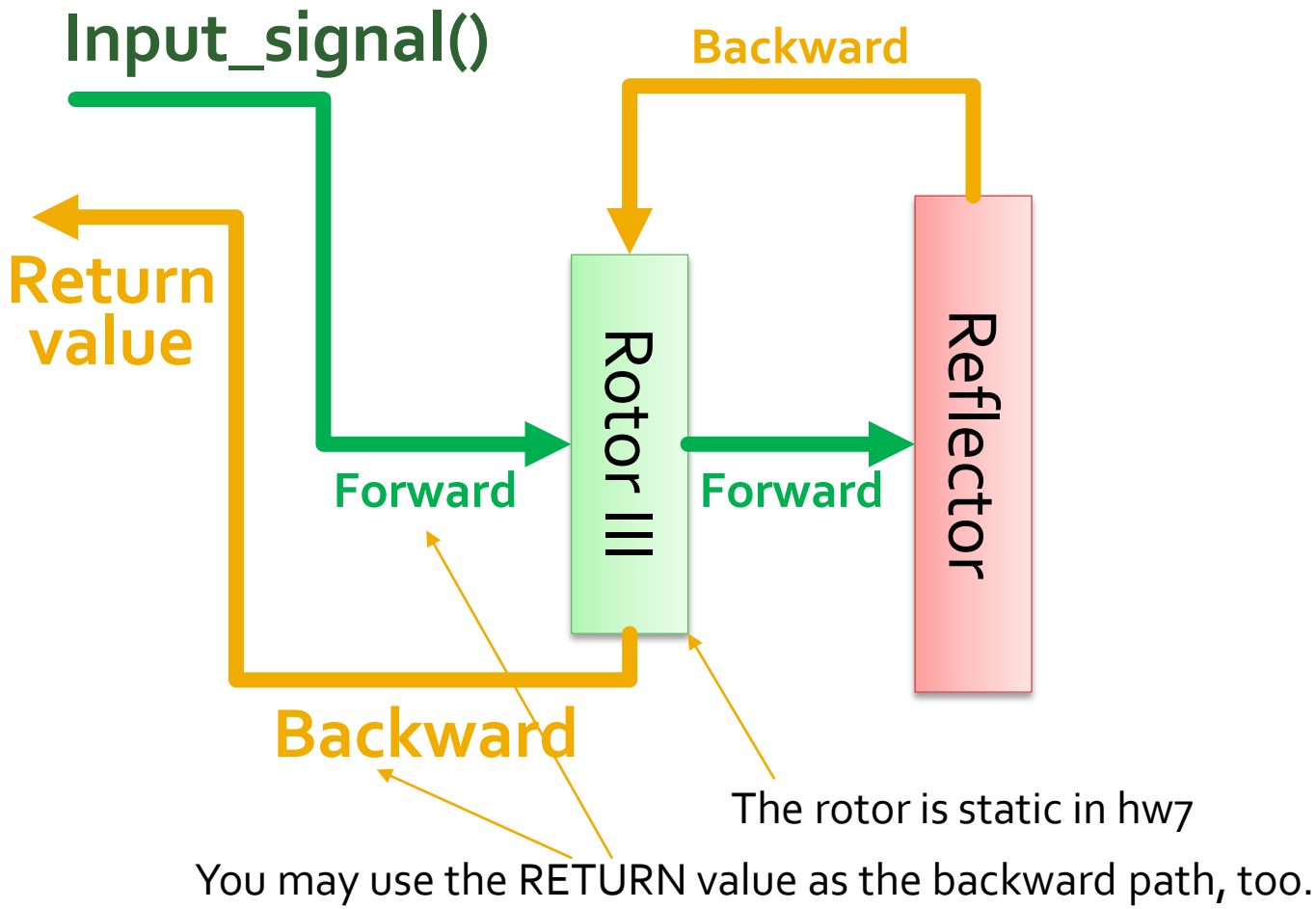
PKRFLGFFYFFFFFFFFFFFFFFFFFFFFXTPMFBFRFFFIFFFFF

Output:

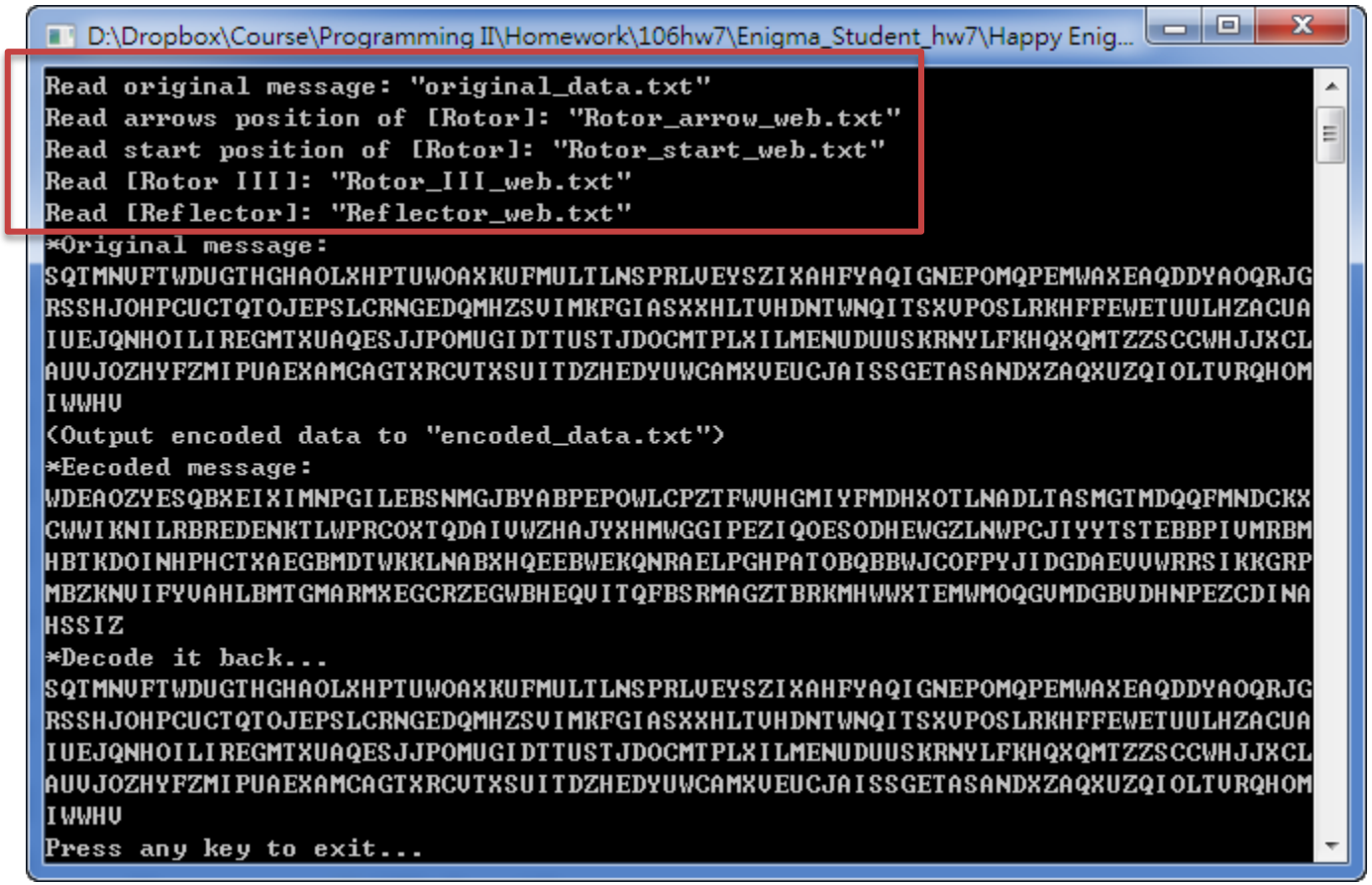
KTWREEOSTUVDOPLLBCOSVXVOLPBZSQKWENMRQHXJWRZDCY

Status: Highlighted wires show steps of encryption.

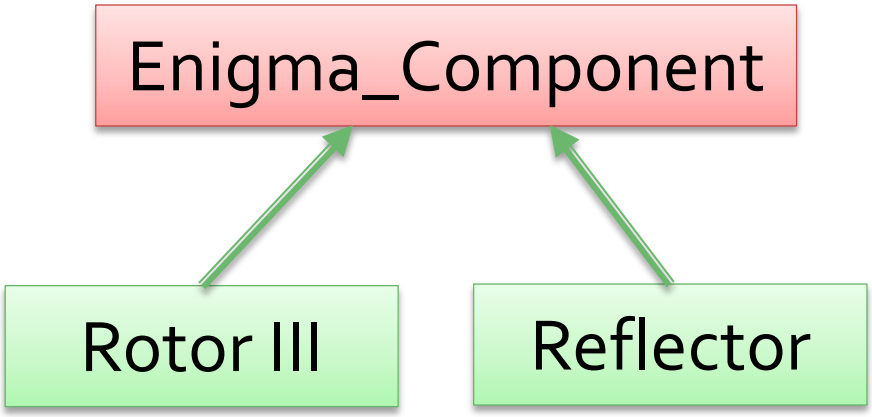
Hw7's Structure (simplifed)



DEMO Time!


D:\Dropbox\Course\Programming II\Homework\106hw7\Enigma_Student_hw7\Happy Enig...
Read original message: "original_data.txt"
Read arrows position of [Rotor]: "Rotor_arrow_web.txt"
Read start position of [Rotor]: "Rotor_start_web.txt"
Read [Rotor III]: "Rotor_III_web.txt"
Read [Reflector]: "Reflector_web.txt"
*Original message:
SQTMNVFTWDUGTHGHAOLXHPTUWOAXKUFMULTLNSPRLVEYSZIXAHFYAQIGNEPOMQPEMWAXEAQDDYAOQRJG
RSSHJOHPCUCTQTOJEPSLCRNGEDQMHZSVIMKFGIASXXHLTVHDNTWNQITSXVPOSLRKHFFEWETUULHZACUA
IUEJQNHOILIREGMTXUAQESJJPOMUGIDTTUSTJDOCMTPLXILMENUDUUSKRNYLFKHQXQMTZZSCCWHJJXCL
AUVJOZHYFZMIPUAEXAMCAGTXRCVTXSUITDZHEDYUWCAMXVEUCJAISSGETASANDXZAQXUZQIOLTVRQHOM
IWWHV
(Output encoded data to "encoded_data.txt")
*Eecoded message:
WDEAOZYESQBXEIXIMNPGILEBSNMGJBYABPEPOWLCPZTFWVHGMIYFMDHXOTLNADLTASMGTMDQQFMNDCKX
CWWIKNILRBREDENKTLWPRCOXTQDAIVWZHAJYXHMWGGIPEZIQOESODHEWGZLNWPCJIYYTSTEBBPIVMRBM
HBTKDOINHPHCTXAEGBMDTWKKLNABXHQEEBWEKQNRAELPGHPATOBQBBWJCOFPYJIDGDAEVVWRRSIKKGRP
MBZKNVIFYVAHLBMTGMARMXEGCRZEGWBHEQVITQFBSRMAGZTBRKMHWWXTEMWMOQGVMDGBVDHNPEZCDINA
HSSIZ
*Decode it back...
SQTMNVFTWDUGTHGHAOLXHPTUWOAXKUFMULTLNSPRLVEYSZIXAHFYAQIGNEPOMQPEMWAXEAQDDYAOQRJG
RSSHJOHPCUCTQTOJEPSLCRNGEDQMHZSVIMKFGIASXXHLTVHDNTWNQITSXVPOSLRKHFFEWETUULHZACUA
IUEJQNHOILIREGMTXUAQESJJPOMUGIDTTUSTJDOCMTPLXILMENUDUUSKRNYLFKHQXQMTZZSCCWHJJXCL
AUVJOZHYFZMIPUAEXAMCAGTXRCVTXSUITDZHEDYUWCAMXVEUCJAISSGETASANDXZAQXUZQIOLTVRQHOM
IWWHV
Press any key to exit...

Hw7's Structure

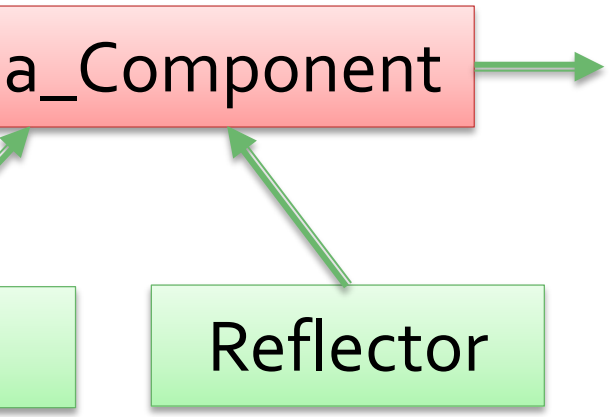


Enigma_Files

Read_file (std::string file)
Write_file (std::string file)
Push (char const& c)
char const& operator [] (size_t const& i)
std::string Data ()
size_t Length ()

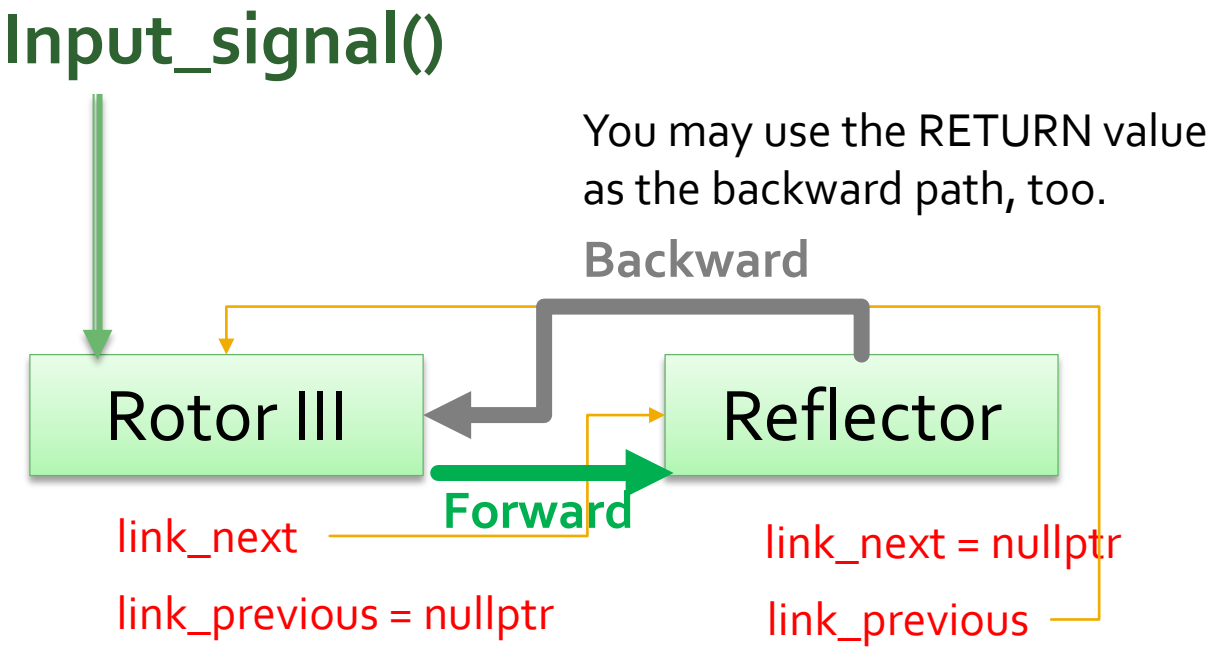
protected:
std::string data ;

Hw7's Structure




```
Enigma_Component ( std::string file )
char Input_signal ( char c )
void Link ( Enigma_Component & next )
virtual void Reset ()

Enigma_Component* link_next ;
Enigma_Component* link_previous ;
//Mapping of forward/backward Encoding
std::vector<size_t> encode_table;
(Optional) std::vector<size_t> reverse_table;
virtual size_t Forward( size_t const& i ) = 0 ;
(Optional) virtual size_t Backward( size_t const& i ) = 0 ;
void Read_table ( std::string file ) ;
virtual void Spin () {}
```

Hw7's Structure



main_106hw7.cpp (Loading)

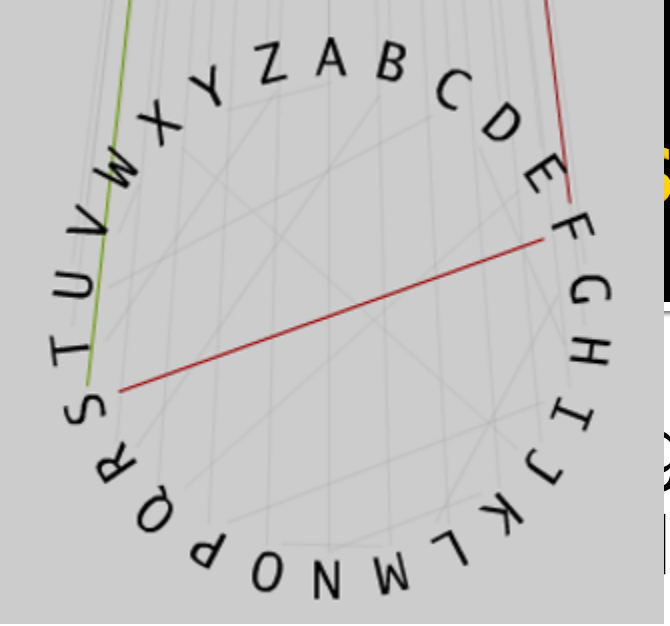
```
Enigma_Files Rotor_start_file, …, encoded_message_file;
…
original_message_file.Read_file("original_data.txt");

Rotor_arrow_file.Read_file(“Rotor_arrow_web.txt”);          //(hw8, part2 only)
Rotor_start_file.Read_file("Rotor_start_web.txt");          //(hw8, part2 only)

Rotor Rotor_III("Rotor_III_web.txt", Rotor_start_file[0], Rotor_arrow_file[0]);
Reflector reflector("Reflector_web.txt");

Rotor_III.Link(reflector);
```

main_106hw7.cpp (Encoding)

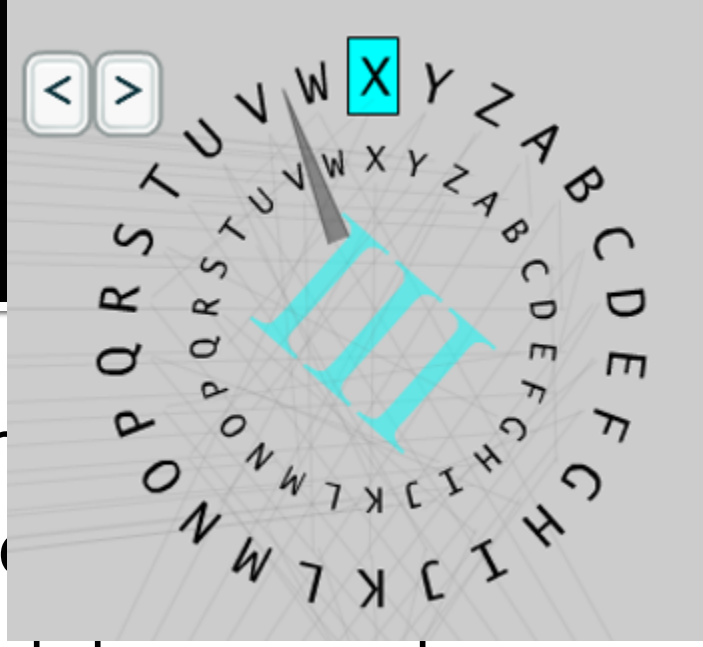
```
std::cout << "*Original message: " << original_message_file.Data() << std::endl;

for (size_t i = 0; i < original_message_file.Length(); i++)
   encoded_message_file.Push(Rotor_III.Input_signal(original_message_file[i]));

encoded_message_file.Write_file("encoded_data.txt");
std::cout << "*Eecoded message: " << encoded_message_file.Data() << std::endl;

std::cout << "*Decode it back..." << std::endl;
Rotor_III.Reset(); //(make a chain reaction to reset all encoder)

for (size_t i = 0; i < encoded_message_file.Length(); i++)
  std::cout << Rotor_III.Input_signal(encoded_message_file[i]);
```

Rules of Assignment

- Encode string from file (original_data.txt) and output encoded string to encoded_data.txt
  - Implement input mechanism to take parameters. (class **Enigma_Files**), for example:
    - Reflector (Reflector_web.txt)
    - Rotor's start positions (Rotor_start_web.txt)
    - Rotor's arrow positions (Rotor_arrow_web.txt)
    - Ringstellung (Rotor_III_web.txt)
- Use **Linked List** to construct the encoding chain and Reset() chain



ignment

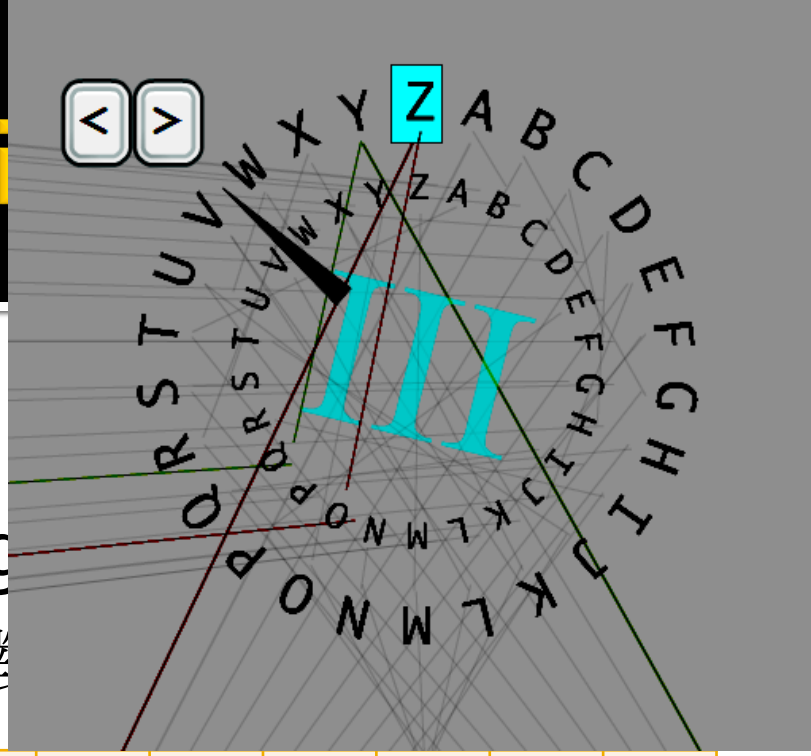


g from file (origir

led string to enc

- Implement input mechanism to take parameters. (class **Enigma_Files**)
  - Reflector (YRUHQSLDPXNGOKMIEBFZCWVJAT)
  - Rotor's start positions (XDH)
  - Rotor's arrow positions (WFR)
  - Ringstellung (BDFHJLCPRTXVZNYEIWGAKMUSQO)
- Use **Linked List** to construct the encoding chain

Ring Settings of Eni



- 字母環 Ring settings (Ring
  - 字母環與旋轉盤線路的相對

Rotor III（外圈）	**A**	**B**	**C**	**D**	**E**	**F**	**G**	**H**	**I**	**J**	**K**	**L**	**M**
Rotor III（內圈）	B	D	F	H	J	L	C	P	R	T	X	V	Z
Rotor III（外圈）	**N**	**O**	**P**	**Q**	**R**	**S**	**T**	**U**	**V**	**W**	**X**	**Y**	**Z**
Rotor III（內圈）	N	Y	E	I	W	G	A	K	M	U	S	Q	O

Reflector of Enigma

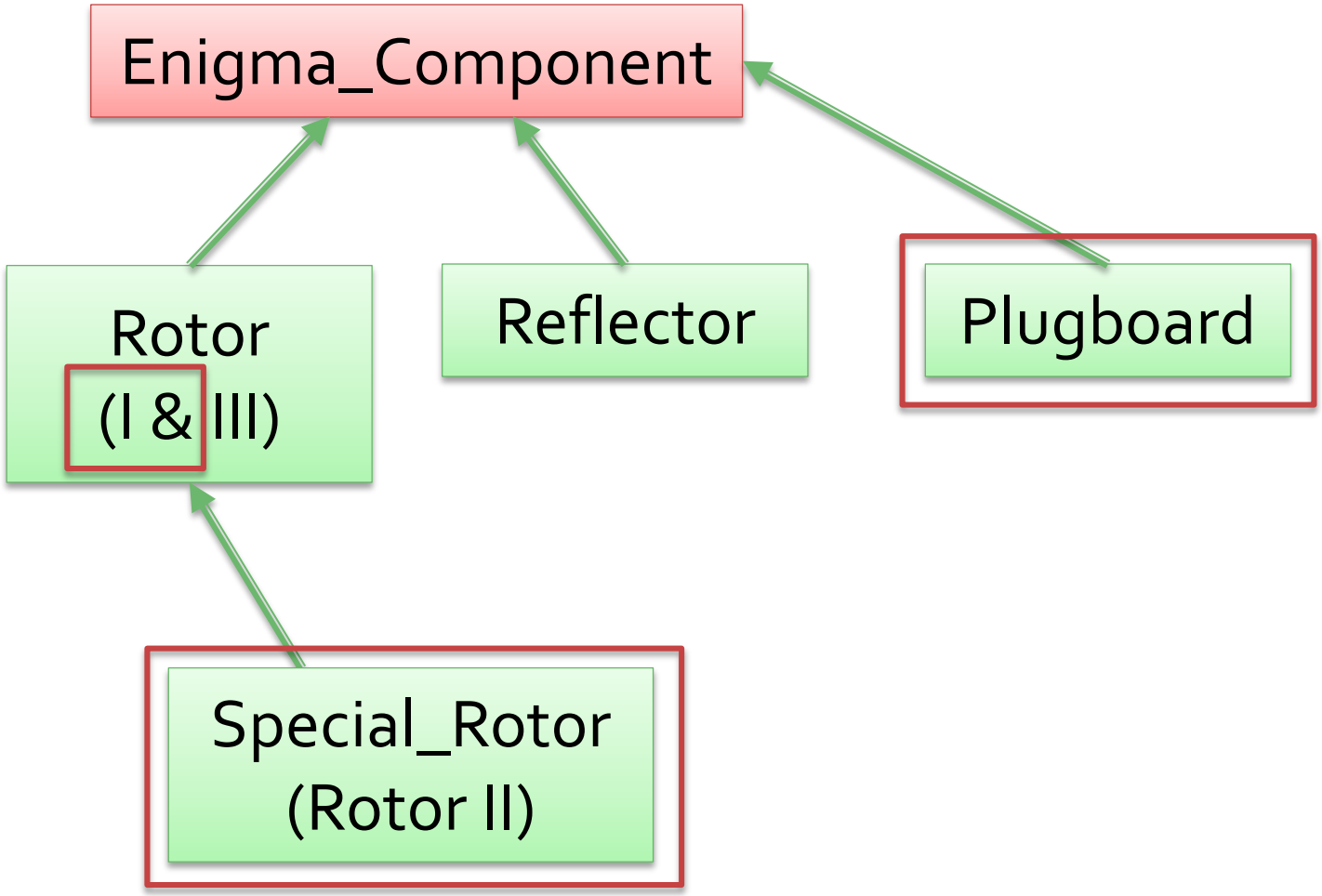
- 反射器 Reflector

Alphabet	**A**	**B**	**C**	**D**	**E**	**F**	**G**	**H**	**I**	**J**	**K**	**L**	**M**
Reflector	Y	R	U	H	Q	S	L	D	P	X	N	G	O
Alphabet	**N**	**O**	**P**	**Q**	**R**	**S**	**T**	**U**	**V**	**W**	**X**	**Y**	**Z**
Reflector	K	M	I	E	B	F	Z	C	W	V	J	A	T

Hw8:
the Enigma
Part II

https://en.wikipedia.org/wiki/Enigma_machine
https://en.wikipedia.org/wiki/Enigma_rotor_details
http://www.enigmaco.de/enigma/ <- DEMO
Due date: 6/17 (Sun.)

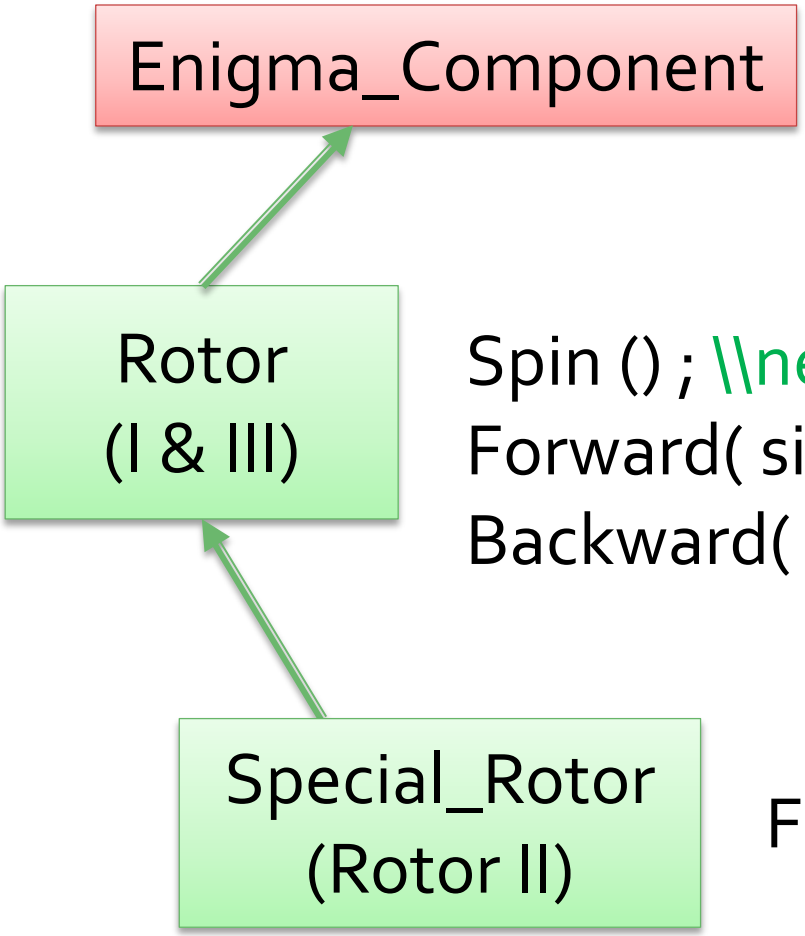
Class Enigma_Component (extended)



Class Enigma_Component (unchanged)


```
Enigma_Component ( std::string file )
char Input_signal ( char c )
void Link ( Enigma_Component & next )
virtual void Reset ()

Enigma_Component* link_next ;
Enigma_Component* link_previous ;
//Mapping of forward/backward Encoding
std::vector<size_t> encode_table;
std::vector<size_t> reverse_table;
virtual size_t Forward( size_t const& i ) = 0 ;
virtual size_t Backward( size_t const& i ) = 0 ;
void Read_table ( std::string file ) ;
virtual void Spin () {}
```

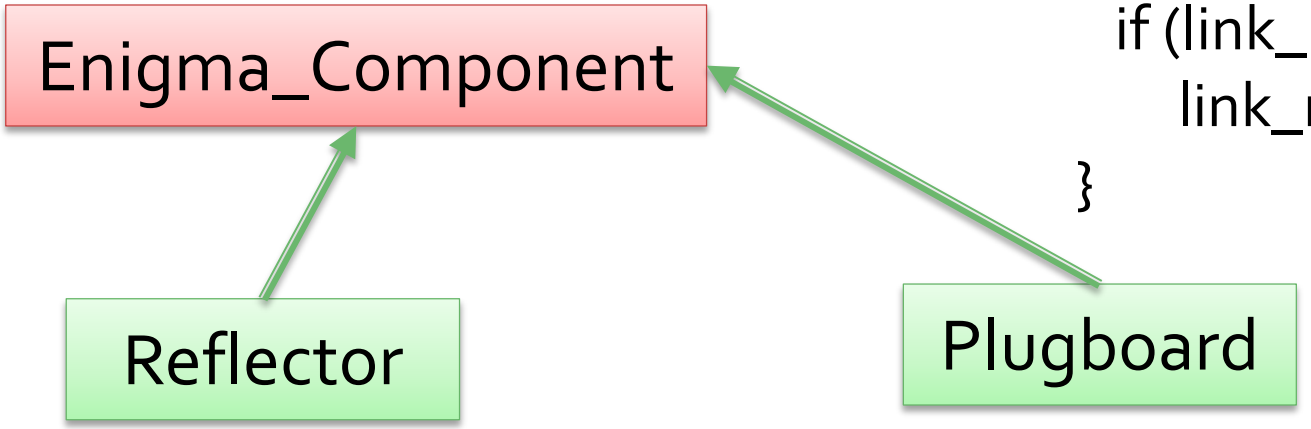
Class Rotor and Special Rotor (extended)



Spin () ; \\new added
Forward(size_t const& i) ; \\updated
Backward(size_t const& i) ;

Forward(size_t const& i) ;

Class Reflector and Plugboard (extended)




```
void Plugboard::Spin(){
  if (link_next)
      link_next->Spin();
}
```

Forward(size_t const& i) ;
Backward(size_t const& i) ;
\\unchanged

Spin();
Forward(size_t const& i) ;
Backward(size_t const& i) ;

main_105hw7.cpp (Loading)

```
Enigma_Files Rotor_start_file, ..., encoded_message_file;
...
original_message_file.Read_file("original_data.txt");
Rotor_arrow_file.Read_file("Rotor_arrow_web.txt");
Rotor_start_file.Read_file("Rotor_start_web.txt");

Plugboard plugboard("Plugboard_web.txt");
Rotor Rotor_III("Rotor_III_web.txt", Rotor_start_file[0], Rotor_arrow_file[0]);
Special_Rotor Rotor_II("Rotor_II_web.txt", Rotor_start_file[1], Rotor_arrow_file[1]);
Rotor Rotor_I("Rotor_I_web.txt", Rotor_start_file[2], Rotor_arrow_file[2]);
Reflector reflector("Reflector_web.txt");

plugboard.Link(Rotor_III);
Rotor_III.Link(Rotor_II);
Rotor_II.Link(Rotor_I);
Rotor_I.Link(reflector);
```

main_106hw8.cpp (Encoding)

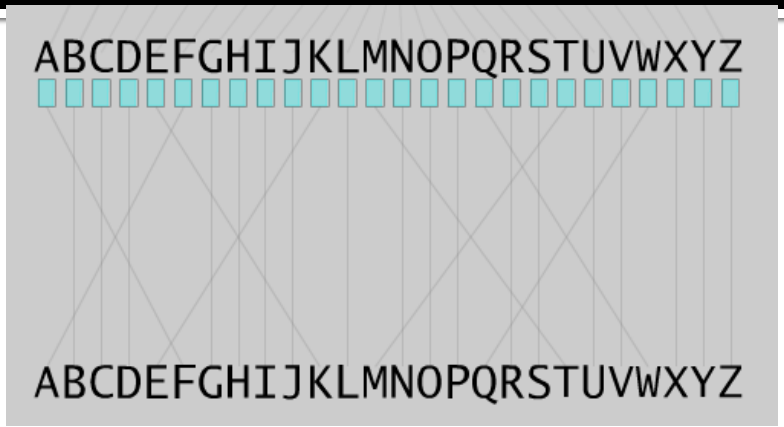
```
std::cout << "*Original message: " << original_message_file.Data() << std::endl;

for (size_t i = 0; i < original_message_file.Length(); i++)
    encoded_message_file.Push(plugboard.Input_signal(original_message_file[i]));

encoded_message_file.Write_file("encoded_data.txt");
std::cout << "*Eecoded message: " << encoded_message_file.Data() << std::endl;

std::cout << "*Decode it back..." << std::endl;
plugboard.Reset(); //(make a chain reaction to reset all encoder)

for (size_t i = 0; i < encoded_message_file.Length(); i++)
   std::cout << plugboard.Input_signal(encoded_message_file[i]);
```

Rules of Assignment

- Encode string from file (original_data.txt) and output encoded string to encoded_data.txt
  - Implement several additional class for different encoder
    - **Rotor** (hw6) for Rotor III & Rotor I **(with spin())**
    - **Special_Rotor** for Rotor_II
    - **Plugboard**
  - Use **Linked List** to construct the encoding chain and Reset() chain

Plugboard of Enigma

- 接線板 Plugboard
  - 接線板的連線



Keyboard	**A**	**B**	**C**	**D**	**E**	**F**	**G**	**H**	**I**	**J**	**K**	**L**	**M**
Plugboard	F	B	C	D	K	A	G	H	I	J	E	L	T
Keyboard	**N**	**O**	**P**	**Q**	**R**	**S**	**T**	**U**	**V**	**W**	**X**	**Y**	**Z**
Plugboard	N	O	P	W	R	S	M	U	V	Q	X	Y	Z

Ring Settings of Enigma

- 字母環 Ring settings (Ringstellung)
  - 字母環與旋轉盤線路的相對位置

Rotor III（外圈）	**A**	**B**	**C**	**D**	**E**	**F**	**G**	**H**	**I**	**J**	**K**	**L**	**M**
Rotor III（內圈）	B	D	F	H	J	L	C	P	R	T	X	V	Z
Rotor III（外圈）	**N**	**O**	**P**	**Q**	**R**	**S**	**T**	**U**	**V**	**W**	**X**	**Y**	**Z**
Rotor III（內圈）	N	Y	E	I	W	G	A	K	M	U	S	Q	O

Ring Settings of Enigma

- 字母環 Ring settings (Ringstellung)
  - 字母環與旋轉盤線路的相對位置

Rotor II（外圈）	**A**	**B**	**C**	**D**	**E**	**F**	**G**	**H**	**I**	**J**	**K**	**L**	**M**
Rotor II（內圈）	A	J	D	K	S	I	R	U	X	B	L	H	W
Rotor II（外圈）	**N**	**O**	**P**	**Q**	**R**	**S**	**T**	**U**	**V**	**W**	**X**	**Y**	**Z**
Rotor II（內圈）	T	M	C	Q	G	Z	N	P	Y	F	V	O	E

Ring Settings of Enigma

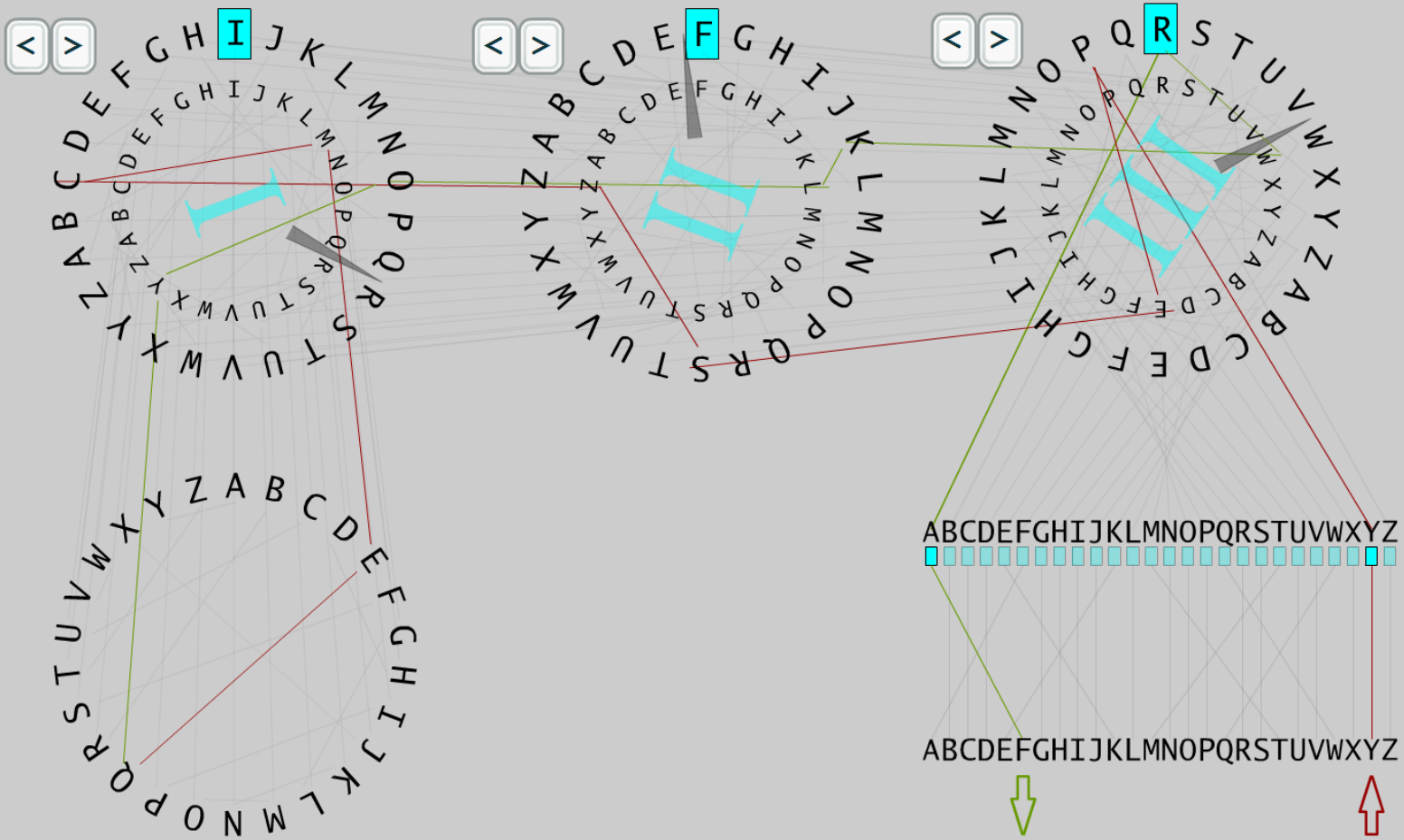
- 字母環 Ring settings (Ringstellung)
  - 字母環與旋轉盤線路的相對位置

Rotor I（外圈）	**A**	**B**	**C**	**D**	**E**	**F**	**G**	**H**	**I**	**J**	**K**	**L**	**M**
Rotor I（內圈）	E	K	M	F	L	G	D	Q	V	Z	N	T	O
Rotor I（外圈）	**N**	**O**	**P**	**Q**	**R**	**S**	**T**	**U**	**V**	**W**	**X**	**Y**	**Z**
Rotor I（內圈）	W	Y	H	X	U	S	P	A	I	B	R	C	J

Reflector of Enigma

- 反射器 Reflector

Alphabet	**A**	**B**	**C**	**D**	**E**	**F**	**G**	**H**	**I**	**J**	**K**	**L**	**M**
Reflector	Y	R	U	H	Q	S	L	D	P	X	N	G	O
Alphabet	**N**	**O**	**P**	**Q**	**R**	**S**	**T**	**U**	**V**	**W**	**X**	**Y**	**Z**
Reflector	K	M	I	E	B	F	Z	C	W	V	J	A	T



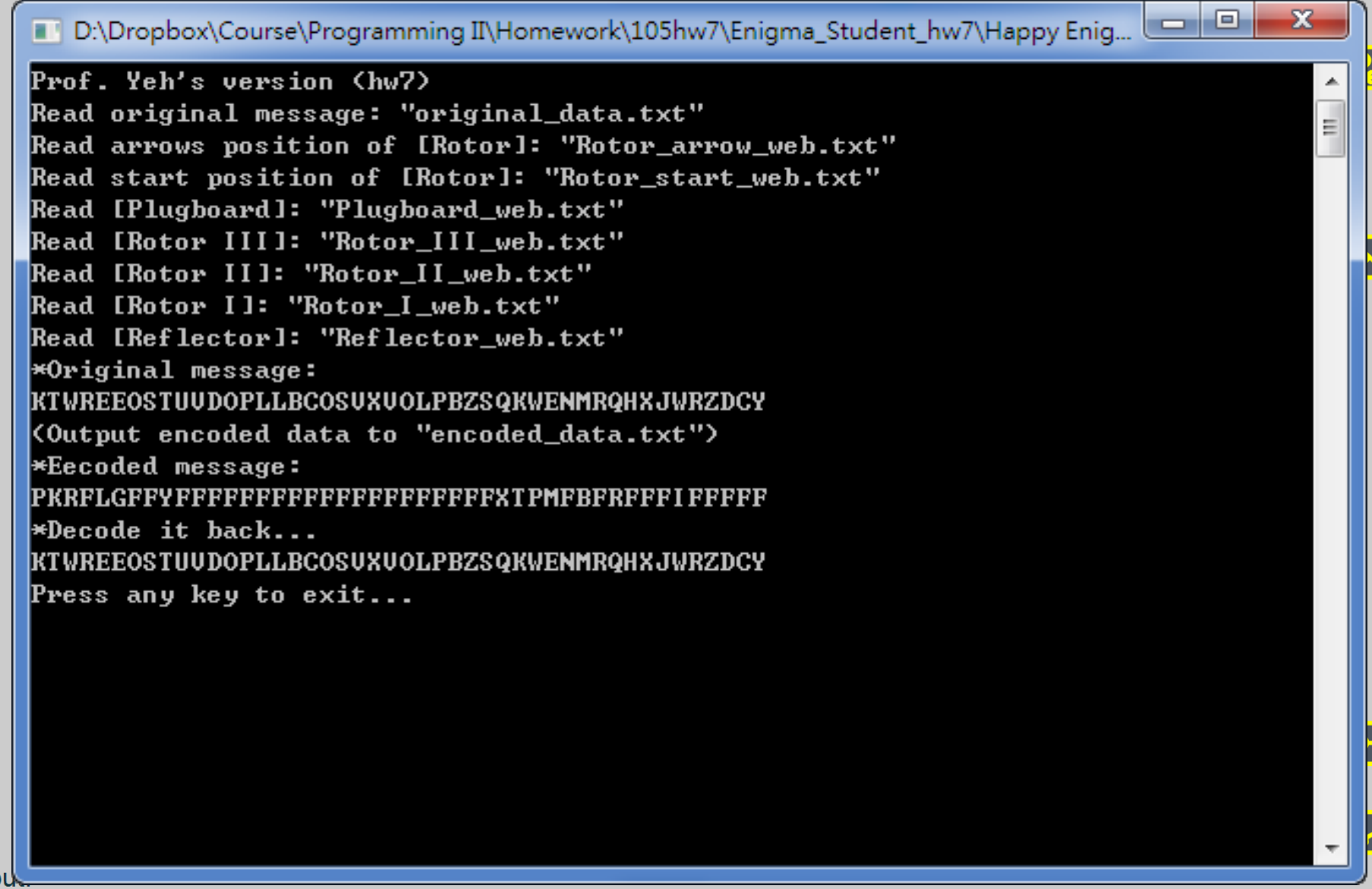
Input:

KTWREEOSTUVDOPLLBCOSVXVOLPBZSQKWENMRQHXJWRZDCY

Output:

PKRFLGFFYFFFFFFFFFFFFFFFFFFFXTPMFBFRFFFIFFFFF

Status: Highlighted wires show steps of encryption.



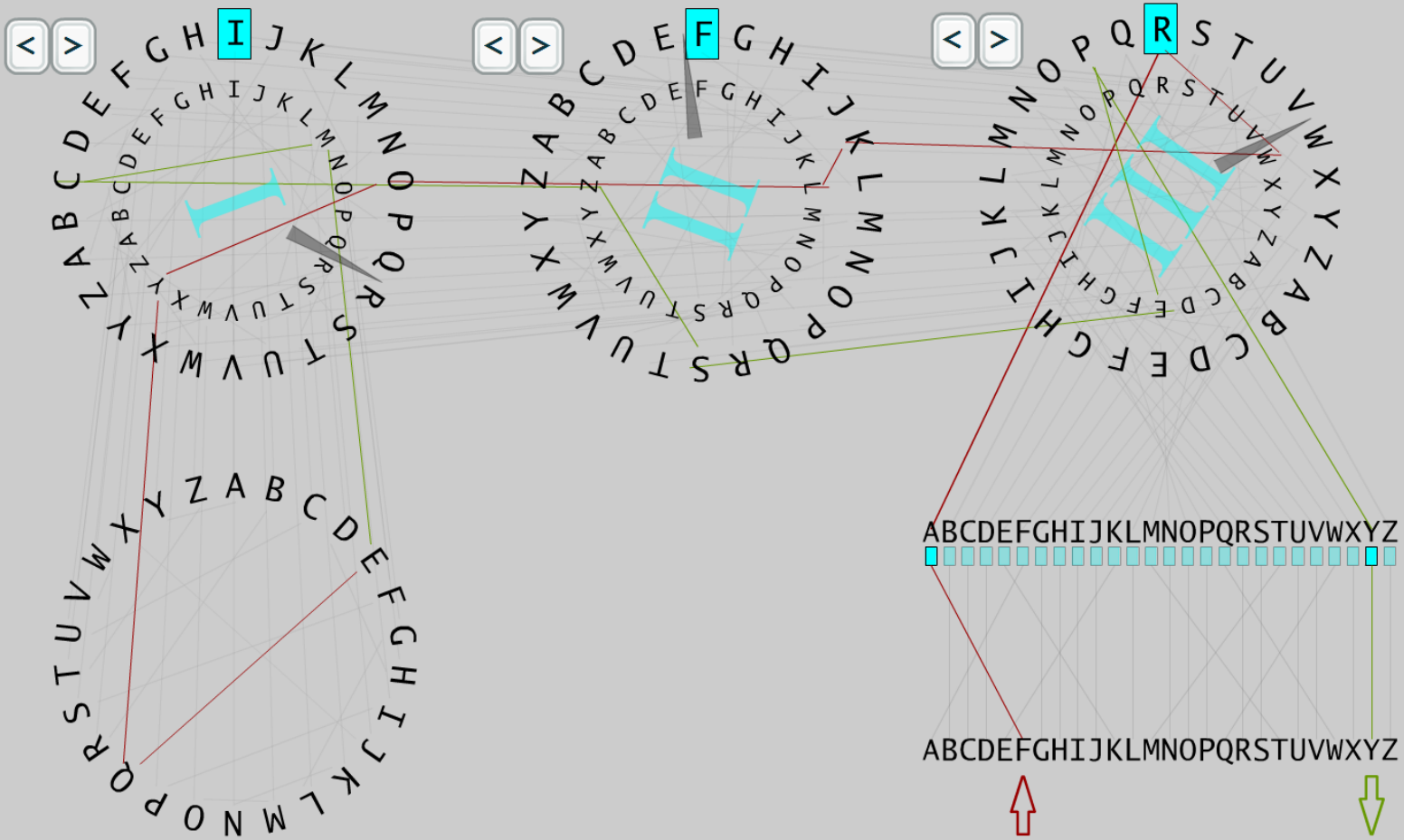
Input:

KTWREEOSTUVDOPLLBCOSVXVOLPBZSQKWENMRQHXJWRZDCY

Output:

PKRFLGFFYFFFFFFFFFFFFFFFFFFFXTPMFBFRFFFIFFFFF

Status: Highlighted wires show steps of encryption.



Reverse

Input:

PKRFLGFFYFFFFFFFFFFFFFFFFFFFFXTPMFBFRFFFIFFFFF

Output:

KTWREEOSTUVDOPLLBCOSVXVOLPBZSQKWENMRQHXJWRZDCY

Status: Highlighted wires show steps of encryption.