Mahir's Code

```
package main

// Rotor represents a single rotor in the Enigma
machine
type Rotor struct {
    name            string // Rotor name (e.g., "I",
"II", "III", etc.)
    wiring          string // Rotor wiring
configuration
    turnover        byte   // Turnover position where
the next rotor will move
    currentPosition byte   // Current position of the
rotor
}

// RotorSet represents a set of interconnected rotors
in the Enigma machine
type RotorSet struct {
    rotors []*Rotor // List of interconnected rotors
}
```

Brenden's Code

```
// Plugboard represents the plugboard in the Enigma
machine
type Plugboard struct {
    pairs map[byte]byte // Letter swaps (e.g., 'A' ->
'Z', 'B' -> 'R', etc.)
}

// Reflector represents the fixed reflector in the
Enigma machine
type Reflector struct {
    wiring [26]int // Reflector wiring configuration
}

// InputRotor represents the input rotor in the Enigma
```

```
machine
type InputRotor struct {
    wiring string // InputRotor wiring configuration
}
```

Yuyao's Code

```
// EnigmaMachine represents the complete Enigma machine
type EnigmaMachine struct {
    plugboard       Plugboard
    inputRotor      InputRotor
    rotorSet        RotorSet
    reflector       Reflector
    currentPosition byte // Current position of the
rotor set (enables manual setting of the initial
position)
}

func main() {
    // Code for main function goes here
}
```