```
package main

import (
        "fmt"
        "strconv"
        "strings"
)

func EncryptThis(text string) string {
        if len(text) == 0 {
                return "String is empty."
        }

        words := strings.Split(text, " ")
        var encryptedWords []string

        for _, word := range words {
                if len(word) == 0 {
                        continue
                }

                // Get ASCII code of the first character
                asciiCode := strconv.Itoa(int(word[0]))

                // Build the encrypted word with ASCII code as the start
                encryptedWord := asciiCode

                // If the word has more than two characters, swap the second and
last
                runes := []rune(word)
                if len(runes) > 2 {
                        encryptedWord += string(runes[len(runes)-1]) +
string(runes[2:len(runes)-1]) + string(runes[1])
                } else if len(runes) == 2 {
                        // If only two characters, add the last character directly
                        encryptedWord += string(runes[1])
                }

                encryptedWords = append(encryptedWords, encryptedWord)
        }

        return strings.Join(encryptedWords, " ")
}

func main() {
        str := "hello world"
        fmt.Println(EncryptThis(str)) // Expected output: "104olle 119drlo"
}
```

```
package main

import (
        "testing"
)

func TestEncryptThis(t *testing.T) {
        tests := []struct {
                input    string
                expected string
        }{
                {"hello world", "104olle 119drlo"}, // Normal case
                {"", "String is empty."},           // Empty string case
                {"a", "97"},                        // Single character case
                {"ab", "97b"},                      // Two characters case
                {"abc", "97cba"},                   // Three characters case
                {"abcd", "97dcb"},                  // Four characters case
                {"abcde", "97edcba"},               // Five characters case
                {"a b c", "97 98 99"},              // Multiple single characters
                {"hello  ", "104olle 104"},         // Trailing spaces
                {"  hello", "104olle"},             // Leading spaces
                {"   ", ""},                        // Only spaces
                {"a b c d e", "97 98 99 100 101"},  // Multiple single characters
                {"test case", "116tase 99e"},       // Normal case with two words
        }

        for _, test := range tests {
                result := EncryptThis(test.input)
                if result != test.expected {
                        t.Errorf("EncryptThis(%q) = %q; expected %q", test.input,
result, test.expected)
                }
        }
}
```