

# 前測

## 測驗內容

以下測驗內容請使用 JavaScript ES5+ 實作。

- 實作 **Fibonacci number** (費式數列)

建立函式 fibonacci 代入參數 position, position 表示的是想要得到 fibonacci sequence 中的第幾個數字的值。

```
fibonacci(0) // 0  
fibonacci(1) // 1  
fibonacci(2) // 1  
fibonacci(3) // 2  
fibonacci(4) // 3
```

- 使用 **Linked List** 實作 **Stack**

實作需包含以下方法。

**push()** : 添加新元素。

**pop()** : 移除元素並返回被移除的元素。

**size()** : 返回所有元素數量。

```
const myStack = new Stack()  
myStack.push(1)  
myStack.push(2)  
myStack.push(3)  
myStack.pop() // 3  
myStack.size() // 2
```

- 實作 **Data Transformer**

將下列輸入資料整合成期望的輸出結果。

輸入資料：

```
const userIds = ['U01', 'U02', 'U03']  
const orderIds = ['T01', 'T02', 'T03', 'T04']  
const userOrders = [  
    { userId: 'U01', orderIds: ['T01', 'T02'] },  
    { userId: 'U02', orderIds: [] },  
    { userId: 'U03', orderIds: ['T03'] },  
]  
const userData = { 'U01': 'Tom', 'U02': 'Sam', 'U03': 'John' }  
const orderData = {  
    'T01': { name: 'A', price: 499 },  
    'T02': { name: 'B', price: 599 },  
    'T03': { name: 'C', price: 699 },  
    'T04': { name: 'D', price: 799 },  
}
```

輸出結果：

```
const result = [
  {
    user: { id: 'U01', name: 'Tom' },
    orders: [
      { id: 'T01', name: 'A', price: 499 },
      { id: 'T02', name: 'B', price: 599 },
    ],
  },
  ...,
]
```

- 擇一實作 **Debounce** 或 **Throttle**

*debounce* 是在 *delay* 時間內如果重新觸發會取消前一次並保留當下觸發的執行。

*throttle* 在觸發後的 *timeout* 時間內只會執行一次。

建立函式 *debounce* 或 *throttle* 帶入參數如下範例：

```
const debounceFunc = debounce(func, delay)
```

```
const throttleFunc = throttle(func, timeout)
```

- (加分題) 使用 **Event Loop** 結合實際操作範例擇一敘述 **Debounce** 或 **Throttle** 的運作方式

如文字輸入、scroll 操作與 button 連續點擊，或是其他可結合 *Debounce* 或 *Throttle* 的行為都可以拿來當作操作範例。

- (加分題) 實作 **React Custom hook**

使用 Create React App 架設，請依照下列登入畫面範例完成 **userForm** 實作。

當有 errors 時 *handleSubmit* 要被 bypass。

...

```
const { handleChange, handleSubmit, values, errors } = useForm({
  initialValues: { account: "", password: "", rememberMe: false },
  validation: (values) => {
    const errors = {}
    if (!values.account) {
      errors.account = "請輸入帳號"
    } else if (!values.password) {
      errors.password = "請輸入密碼"
    }
    return errors
  },
  onSubmit: (values) => console.table(values),
})
```

```
return (
```

```
<>
```

```
  <input name="account" onChange={handleChange} value={values.account}
```

```
placeholder="Account"/>
```

```
  {errors.account && (<div>{errors.account}</div>)}
```

```
<input name="password" onChange={handleChange} value={values.password}
placeholder="password" />
{errors.password && (<div>{errors.password}</div>)}
<label><input type="checkbox" name="rememberMe" onChange={handleChange}
checked={values.rememberMe} />Remember Me</label>
<button onClick={handleSubmit}>Login</button>
</>
)
...
``
```

## 測驗時間

- 請在收到前測 **N+2** 天內完成並回覆。如在時間內無法完成所有測驗也請回覆

## 繳交方式

- 完成後將測驗上傳至個人 github 並回覆 email 且附上 github 連結

附註：

Repo 名稱盡量不要有 AsiaYo 等用詞以防你的努力被抄襲。  
如對測驗有任何問題也可隨時詢問，祝你有美好的一天。