|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  | | --- | --- | --- | |  |  |  | | МИНОБРНАУКИ РОССИИ | | | | Федеральное государственное бюджетное образовательное учреждение  высшего образования  **«МИРЭА** – **Российский технологический университет»**  **РТУ МИРЭА** | | | | |
| Институт ИКБ | |
| Кафедра защиты информации | |

|  |  |
| --- | --- |
| **ОТЧЁТ ПО ПРАКТИЧЕСКОЙ РАБОТЕ № 6** | |
| **по дисциплине «Технологии и методы программирования»** | |
|  | |
| Выполнила студент группы БИСО-02-21 | Одинцова Ю.А. |
| Проверил | Лесько С.А. |

Москва 2023

**Практическая работа №6**

Ссылка:

<https://github.com/just-student-19/tmp_5_sem/tree/практика_6>

## Инверсия контроля

*Листинг программы:*

class Processor {

process(callback) {

console.log("Выполняется обработка...");

callback();

console.log("Обработка завершена.");

}

}

function myCallback() {

console.log("Вызвана функция обратного вызова!");

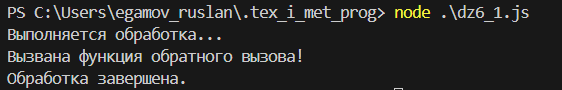
}

// Main

const processor = new Processor();

processor.process(myCallback);

*Скриншот выполнения программы:*



## Заместитель

*Листинг программы:*

class Subject {

request() {

throw new Error('Ошибка субъекта!');

}

}

class Handler extends Subject {

request() {

console.log('Handler: Обрабатываем запрос...');

}

}

class Proxy extends Subject {

constructor() {

super();

this.\_handler = null;

}

request() {

if (this.\_handler === null) {

this.\_handler = new Handler();

}

console.log('Proxy: Действия перед выполнением запроса');

this.\_handler.request();

console.log('Proxy: Доп. действия после выполнения запроса');

}

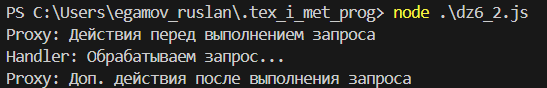
}

// Main

const proxy = new Proxy();

proxy.request();

*Скриншот выполнения программы:*



## Компоновщик

*Листинг программы:*

class Component {

operation() {

throw new Error('Ошибка при выполнении операции!');

}

}

class Notice extends Component {

operation() {

console.log('Notice: выполнение операции.');

}

}

class Alpha extends Component {

constructor() {

super();

this.children = [];

}

add(component) {

this.children.push(component);

}

operation() {

console.log('Alpha: выполнение операции для всех дочерних компонентов.');

for (let child of this.children) {

child.operation();

}

}

}

// Main

const composite = new Alpha();

composite.add(new Notice());

composite.add(new Notice());

composite.operation();

*Скриншот выполнения программы:*

