



1. 아래 네모상자와 같은 것은 무엇일까요?

```
const 객체 = {
```

```
  name: qqk.name,  
  age: qqk.age,  
  school: qqk.school
```

```
}
```

1) _qqq

2) qqk.spread()

3) ...rest

4) ...qqq

2. 다음 중 rest에 담겨있는 값은 무엇인가요?

```
const profile = {  
  name: "철수",  
  age: 12,  
  school: "다람쥐초등학교"  
}
```

```
const { name, ...rest } = profile
```

1) age

2) school

3) age, school을 제외한 나머지

4) age, school

3. 아래 console.log의 결과는 무엇인가요?

```
const profile = {  
  name: "철수",  
  age: 12,  
  school: "다람쥐초등학교",  
  age: 15  
}
```

```
console.log(profile.age)
```

1) 12

2) 15

3) 에러

4) 12, 15

4. product.controller.ts 파일에서 this.productService.qqq()의 결과는 무엇인가요?

```
const productService = new ProductService();  
  
// 상품API  
const productController = new ProductController(productService);  
app.post("/products/buy", productController.buyProduct);
```

index.ts

```
export class ProductController {  
  productService;  
  
  constructor(productService) {  
    this.productService = productService;  
  }  
  
  buyProduct(req, res){  
    this.productService.qqq();  
  };  
}
```

product.controller.ts

```
export class ProductService {  
  qqk() {  
    return "Hello World";  
  }  
}
```

product.service.ts

정답) _____

5. productService가 new ProductController()에 들어가고 있습니다.
이것을 무엇이라고 하나요?

```
const productService = new ProductService();  
  
// 상품API  
const productController = new ProductController(productService);  
app.post("/products/buy", productController.buyProduct);
```

index.ts

```
export class ProductController {  
  productService;  
  
  constructor(productService) {  
    this.productService = productService;  
  }  
  
  buyProduct(req, res){  
    this.productService.qqq();  
  };  
}
```

product.controller.ts

```
export class ProductService {  
  qqk() {  
    return "Hello World";  
  }  
}
```

product.service.ts

정답) _____

6. productService 앞에 private readonly를 추가해 보았습니다.
이렇게 함으로써 몇 번째 줄이 생략되나요? (힌트: 2줄)

```
1 export class ProductController {  
2   productService;  
3  
4   constructor(private readonly productService) {  
5     this.productService = productService;  
6   }  
7  
8   buyProduct(req, res){  
9     this.productService.qqq();  
10  };  
11 }
```

product.controller.ts

정답) 줄번호 ____ 번째 줄
줄번호 ____ 번째 줄

7. 아래의 ProductService 타입은 어떻게 만들 수 있나요? (힌트: 정답 2개)

```
export class ProductController {  
  constructor(private readonly productService: ProductService) {}  
  
  buyProduct(req, res){  
    this.productService.qqq();  
  };  
}
```

product.controller.ts

1) class 사용

2) console.log 사용

3) protected 사용

4) interface 사용

8. 왼쪽은 express 이고, 오른쪽은 nest 입니다.
왼쪽과 같이 오른쪽 nest 에서 의존성주입은 어떻게 하면 될까요?

```
const productService = new ProductService();  
  
// 상품API  
const productController = new ProductController(productService);  
app.post("/products/buy", productController.buyProduct);
```

index.ts

```
@Module({  
  imports: [],  
  controllers: [ProductController],  
  providers: [],  
})  
export class ProductModule {}
```

product.module.ts

정답)

9. 아래 @Module 에서 사용된 @를 무엇이라고 하나요?

```
@Module({  
  imports: [],  
  controllers: [ProductController],  
  providers: [],  
})  
export class ProductModule {}
```

product.module.ts

정답) _____

10. 아래 @Module은 무엇으로 만들 수 있을까요?

```
@Module({  
  imports: [],  
  controllers: [ProductController],  
  providers: [],  
})  
export class ProductModule {}
```

product.module.ts

1) 함수

2) 변수

3) 반복문

4) 조건문

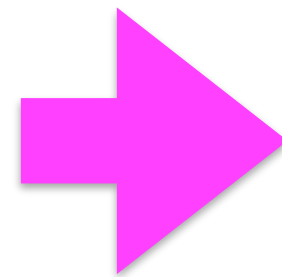
11. 아래 타입에서 name, age, school만 있는 타입을 만들고 싶습니다.
어떻게 하는게 효율적인가요?

```
interface IProfile {  
  name: string;  
  age: number;  
  school: string;  
  hobby?: string;  
}
```

- 1) interface 를 사용하여 name, age, school만 존재하는 타입을 만든다.
- 2) 유틸리티타입의 Pick 타입을 사용한다.
- 3) 유틸리티타입의 Omit 타입을 사용한다.
- 4) 제네릭타입을 사용하고, 타입명을 T, U, V 로 지정한다.

12. 현재 nest 프레임워크에서 그래프큐엘을 code-first 방식으로 사용 중에 있습니다. 왼쪽은 잘못된 DTO 이고, 오른쪽은 결과 타입입니다. 잘못된 DTO를 어떻게 수정하면 오른쪽과 같이 나올까요?

```
@ObjectType()  
export class CreateBoardInput {  
  @Field(() => String)  
  writer: string;  
  
  @Field(() => String)  
  title: string;  
  
  @Field(() => String)  
  contents: string;  
}
```



```
input CreateBoardInput {  
  writer: String!  
  title: String!  
  contents: String!  
}
```

잘못된 DTO

결과

정답)

끝!