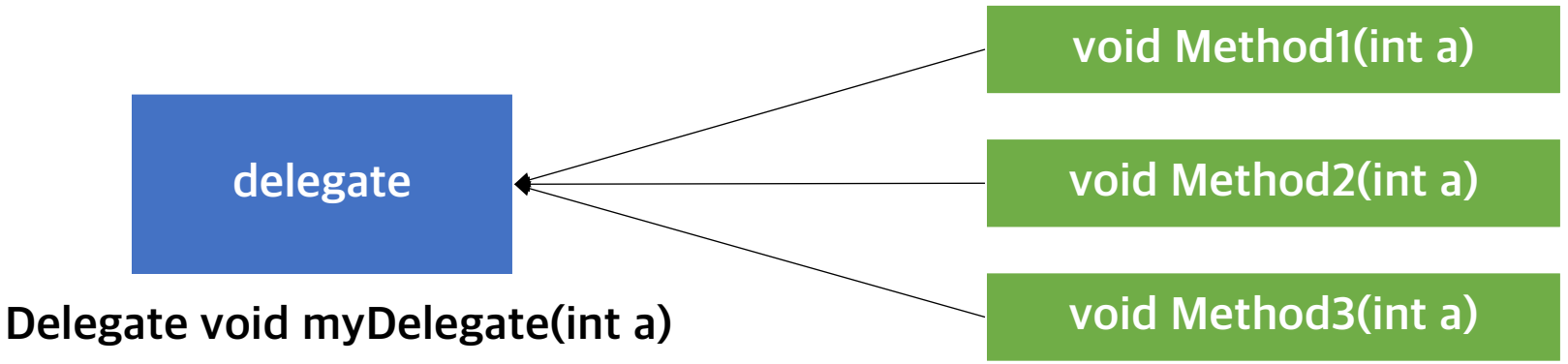
codingOn

Delegate

Delegate

- 메서드에 대한 참조를 저장할 수 있는 형식(타입)
- 함수를 대신 실행해 주는 객체 (간접 실행)
- 함수를 변수처럼 배열로 선언하거나 매개변수(함수의 입력 값)으로 활용
  - 입력 값으로 받은 함수를 실행하는 것을 Callback 이라고 함
- **Delegate**의 입출력 자료형과 **Method**의 것이 같아야 함



Delegate

- 활용 예시

```
delegate void CallDelegate(string msg);

3 references
public partial class Form1 : Form
{
    CallDelegate callDelegate;

    1 reference
    void Print(string msg) => textBox_print.Text = msg;

    1 reference
    public Form1()
    {
        InitializeComponent();

        this.callDelegate = new CallDelegate(Print);
        this.callDelegate("Hello"); // Hi 메소드 실행
    }
}
```

delegate 정의

delegate 인스턴스 선언

delegate 인스턴스에 메서드 저장

delegate를 통한 메소드 호출 및 입력 값 전달

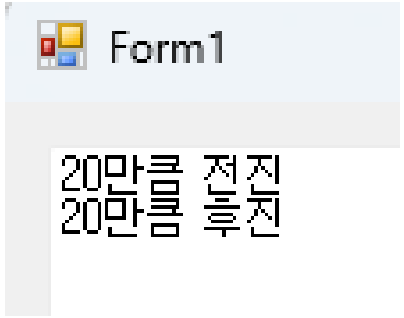
Delegate - multicasting

- 하나의 대리자에 여러개의 함수를 등록하는 것도 가능

```
2 references
internal class Tank
{
    4 references
    public string result {  get; set; } = string.Empty;
    1 reference
    public void Forward(int length) => result += $"{length}만큼 전진\r\n";
    1 reference
    public void Backward(int length) => result += $"{length}만큼 후진\r\n";
    2 references
    public void Rotate(int angle) => result += $"{angle}도 만큼 회전\r\n";
}
```

Delegate

```
// 사용하려는 함수와 입출력 데이터 타입을 맞춤
delegate void Move(int value);

1 reference
public Form1()
{
    InitializeComponent();

    Tank tank = new Tank();

    // += 기호를 사용하여 여러개의 함수를 대리자에 등록
    Move move;
    move = tank.Forward;
    move += tank.Backward;
    move += tank.Rotate;

    // -= 기호를 사용하여 등록된 함수를 제거 가능
    move -= tank.Rotate;
    move(20);

    // 대리자를 통해 입력 값을 전달하는 것도 가능
    textBox_print.Text += tank.result;
}
```



Delegate

- 대리자는 함수의 매개변수로 사용될 수 있음

```
// 대리자 선언
delegate void Runner();
// 대리자를 매개변수로 사용하는 메소드
2 references
void RunnerCall(Runner runner) => runner();
1 reference
void Go() => textBox_print.Text += "달려\r\n";
1 reference
void Stop() => textBox_print.Text += "멈춰\r\n";
```

```
1 reference
public Form1()
{
    InitializeComponent();

    // 대리자를 메소드의 매개변수로 전달
    Runner runner = new Runner(Go);
    RunnerCall(runner);
    // 즉석에서 인스턴스를 생성하여 전달하는 것도 가능
    RunnerCall(new Runner(Stop));
}
```

codingOn

실습. Delegate & 제네릭컬렉션 & 람다식

1. **Product** 클래스 정의
   - 필드(속성): 제품명 (Name), 가격 (Price), 카테고리 (Category)
   - 생성자: 3개의 값 받아서 필드(속성) 초기화
   - ToString() 메서드 오버라이드
     - “제품명 / 가격 / 카테고리“ 형식으로 출력되도록 작성.
2. **ProductFilter** 클래스 정의
   - 델리게이트 정의: ProductCondition
     - Product를 입력받고 bool을 반환하는 함수 형식
   - 정적 메서드 정의:
     - 조건을 만족하는 제품만 새 리스트에 담아 반환.
     - (Hint) public static [반환형] [메소드명]([매개변수1, 매개변수2])
       - 매개변수 1: 전체 제품 목록
       - 매개변수 2: 람다식 (조건)

실습. Delegate & 제네릭컬렉션 & 람다식

3. main 구현

• 제네릭 컬렉션(List<T>)을 이용하여 5개 이상의 임의의 제품 객체 추가

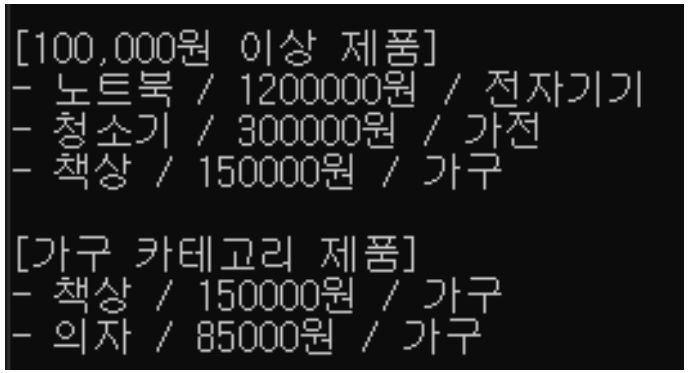
[구현한 정적 메서드 활용]

• 가격이 **10만원 이상**인 제품만 필터링

• 카테고리가 " 가구 " 인 제품만 필터링

• 각각의 결과를 출력 후, 출력 결과 및 Repo. Slack 댓글로 업로드.

<결과화면 예시>



codingOn

Event

- 특정 알림(Event)을 설정하여, 해당 알림이 발생하면 연관된 모든 메소드(Event Subscriber)가 일괄적으로 실행
  - 특수한 형태의 delegate라고 할 수 있음
- EventHandler 라는 객체가 delegate의 역할을 수행
- delegate와 다르게 주로 클래스 내부에 선언
- WinForm, WPF 등에서 마우스 클릭, 마우스 오버, 마우스 아웃, 키보드 입력 등 모두 Event에 해당

Event

```
internal class MyButton
{
    public event EventHandler Click;

    1 reference
    public void MouseButtonDown()
    {
        if(this.Click != null)
        {
            this.Click(this, EventArgs.Empty);
        }
    }
}
```

```
public FormEvent()
{
    InitializeComponent();

    MyButton button = new MyButton();

    button.Click += new EventHandler(ButtonClick1);
    button.Click += new EventHandler(ButtonClick2);

    button.MouseButtonDown();
}

// event에 등록될 메소드는 object sender, EventArgs e를 가져야 함
1 reference
void ButtonClick1(object sender, EventArgs e)
{
    MessageBox.Show("버튼1 클릭됨!");
}

1 reference
void ButtonClick2(object sender, EventArgs e)
{
    MessageBox.Show("버튼2 클릭됨!");
}
```

실습. Delegate & Event

- Delegate를 사용하여 Event 시스템을 구현하기
- **EventDelegate**라는 이름의 delegate를 선언
- EventManager 클래스
  - Event 이름, **EventDelegate**가 각각 Key, Value로 된 Dictionary 생성
  - 이벤트 등록 메소드에서는 Dictionary에 이벤트를 추가
  - 이벤트 삭제 메소드에서는 Dictionary에서 이벤트를 제거
  - 이벤트 실행 메소드에서는 Dictionary 안에 있는 이벤트를 선택하여 Invoke()
- Form 에서 EventManager 클래스의 인스턴스를 생성하고 Form의 메소드들을 이름과 함께 EventManager에 등록
- Push 후 GitHub Repo. URL을 슬랙 댓글로 제출