```java
package ch14;

import java.util.function.Function;
//매서드참조는 람다식을 더 줄요서 표현한 방식이다. 람다식을 잘 이해고 사용하는 것이 중요하다 !
public class Method참조 {

	public static void main(String[] args) {

		/*
		 *  하나의 입력, 반환을 가지는 매서드를 가지고 있다.
		 *  Generic을 사용하면 type정보가 다른 여러개의  Function인터페이스를 만들 필요가 없다.
		 *  사용하는 시점에서  Type을 결정하면된다
		 *
		 *  interface Function<T,R> {
		 *      R  apply ( T t);
		 *  }
		 *
		 */


		//#####################################
		//1.인터페이스를 이름있는 클래스로 구현하고 사용하기
		//이름있는 클래스를 구현
		   class MyFunction implements Function<String, Integer> {
				@Override
				public Integer apply(String t) {
				    return Integer.parseInt(t);
				}
		   }



		//객체생성          및 매서드 호출
		  MyFunction f0 = new MyFunction();
		  int result0 = f0.apply("100");
		  System.out.println( result0);


		 //########################################
		  // 2. 인터페이스를 익명클래스로 만들어서 사용하기
		 //########################################

		Function<String , Integer> f2 = new Function<>() {
			@Override
			public Integer apply(String t) {

				return Integer.parseInt(t);
		}};

		int result2 =f2.apply("24");
		System.out.println( result2);
```

```
        //##########################################
         // 3. 람다식사용하기 : 마치 함수만을 사용하는 것처럼 사용할 수 있다.
        //##########################################

        //람다식은 내부적으로 익명클래스  익명객체가 만들어짐
        //자바는 객체없이는 매서드를 사용할 수 없다.
        //마치 매서드만을 사용하는것처럼 보이지만 내부적으로 클래스와 객체가 만들어진다
        //하나만 매서드를 가진 객체를 만드는 것이다


        Function<String , Integer> f  = ( String s) -> Integer.parseInt(s);
        int result  =f.apply("12");
        System.out.println(result);




        //##########################################
        // 4. 매서드참조사용하기 : 람다식을 더 줄여서 표현한다 ,
        람다식을 작성 후 chatgpt에게 매서드 참조로 바꿔줘!!구경하기
        //##########################################
        //람다식을 더욱 간결하게 표현할 수 있는 방법 : 매서드 참조
        //람다식을 더 줄여서 사용할 수 있다. 매서드 참조라고 한다
        //람다식을 잘 이해하는 것이 중요하다
        //매서드 참조는 눈으로만 익히자

        Function<String , Integer> f3  = Integer::parseInt;
        int result3  =f3.apply("12");
        System.out.println(result3);

    }

}
```