==========
(실습10) Promise 사용법 연습
==========
(환경설정)

본 교과목 실습폴더에서 (실습10) 압축파일(lab10.zip)을 복원하여 <lab10> 폴더에서
터미널 창을 열고 다음 프로그램을 실행한다.

```
> yarn install
```

(실습10)
---
1) "najax_get_final.js"은 자바스크립트 콜백 체인 형태로 구현되어 있으며, 이 프로그램의
복잡도를 줄이기 위하여 자바스크립트 최신문법(ES6, 2015년)에서 채택한 Promise를
사용하여 2개 프로그램(najax_get_promise.js, najax_get_promise_v2.js)을 작성하였다.
이 프로그램들을 이해한 다음, 콜백 체인 형태의 프로그램인 "request_final.js"을
Promise 버전으로 변환한 2개 프로그램(request_promise.js, request_promise_v2.js)을
작성하라.

(*) Promise 사용법을 설명하는 예제로 타임아웃 비동기 호출에 대하여 콜백 체인 방법
구현(setTimeout_callback.js)과 Promise 버전 구현(setTimeout_promise.js,
setTimeout_promise_v2.js)을 제공함

(**) Promise 생성 방법의 일관성 개선을 위하여 작성한 모듈(MyPromise_v2.js)을 사용하여
동기-비동기 실행을 교차적으로 호출하는 예제(sync_async_promise_v2)가 추가 제공됨

2) Promise는 복수개의 비동기처리를 동시 실행하여 실행시간 최적화를 제공하는 all() 함수를
제공하고 있다. 예를 들면, 위에서 사용한 2개 프로그램(najax_get_promise.js,
setTimeout_promise.js)을 병렬처리하는 프로그램(najax_timeout_all_promise.js)에서
all() 함수를 사용한다. Promise 생성 방법의 일관성을 위하여 제공하는 모듈(MyPromise_v2.js)를
사용하여 수정된 병렬처리 프로그램(najax_timeout_all_promise_v2.js)도 제공된다.
이 수정된 병렬처리 프로그램에 대하여, 1)에서 구현한 비동기 처리 기능도 3번째 병렬처리하도록
2개 프로그램(najax_request_timeout_all_promise.js, najax_request_timeout_all_promise_v2.js)을
완성하여 다음의 출력 값이 나오도록 하라.

(najax_timeout_all_promise_v2.js - 실행결과)

```
긴급 작업
 google page received: 작업1 완료!
 google page files saved: 작업2 완료!
 [ 'google page file created', '작업3 완료!' ]
```

(najax_request_timeout_all_promise_v2.js - 실행결과)

```
긴급 작업
google/naver page received: 작업1 완료!
google/naver page files saved: 작업2 완료!
[ 'google page file created', 'naver page file created', '작업3 완료!' ]
```

(결과제출)

다음주 월요일 수업시간 평가 종료 후, 완성프로그램들을 가상대학에 제출
(제출대상) 작성한 모든 파일들을 "lab10_final.zip"으로 압축/업로드