# **Trabajo Práctico: Objetivos**

## **Parte 1: npm y package.json**

### **1. Introducción a npm**

**npm (Node Package Manager)** es el gestor de paquetes de Node.js, utilizado para instalar, compartir y administrar librerías y herramientas en proyectos de JavaScript.

### **2. Creación de un proyecto con npm**

#### **Pasos:**

1. Abre una terminal y crea una nueva carpeta para el proyecto:  
   mkdir mi-proyecto && cd mi-proyecto
2. Inicializa un proyecto de Node.js con:  
   npm init o npm init -y

### **3. Explorando package.json**

El archivo package.json contiene metadatos del proyecto y dependencias. Ejemplo de un archivo package.json:

{

"name": "mi-proyecto",

"version": "1.0.0",

"description": "Un proyecto de prueba",

"main": "index.js",

"scripts": {

"start": "node index.js"

},

"dependencies": {

"yargs": "^17.7.2"

}

}

### **4. Instalación de paquetes**

Para instalar un paquete, usa:

npm install nombre-del-paquete

Por ejemplo, para instalar yargs:

npm install yargs

Para instalar dependencias de desarrollo:

npm install eslint --save-dev

## **Parte 2: Uso de Yargs**

### **1. Introducción a Yargs**

**Yargs** es una librería que facilita el manejo de argumentos en la línea de comandos en Node.js.

### **2. Creación de un script con Yargs**

Crea un archivo index.js y escribe el siguiente código:

const yargs = require('yargs');

const argv = yargs

.command('saludar', 'Muestra un saludo', {

nombre: {

describe: 'Nombre de la persona a saludar',

demandOption: true,

type: 'string'

}

})

.help()

.argv;

if (argv.\_.includes('saludar')) {

console.log(`Hola, ${argv.nombre}!`);

}

### **3. Ejecutando el script**

Ejecuta el siguiente comando en la terminal:

node index.js saludar --nombre="Juan"

Salida esperada:

Hola, Juan!

## **Ejercicios adicionales**

### **Ejercicios teóricos:**

1. Explica qué es npm y su función en un proyecto de Node.js.
2. Describe la estructura básica del archivo package.json y menciona tres campos importantes.
3. ¿Cuál es la diferencia entre dependencias normales y dependencias de desarrollo en npm?
4. ¿Qué comando utilizamos para actualizar un paquete instalado en nuestro proyecto?
5. ¿Para que se utiliza el comando npm link?
6. Ver el módulo de inicio de un proyecto

### **Ejercicios prácticos:**

1. Agrega otro comando a Yargs llamado despedir que muestre "Adiós, [nombre]!".
2. Crea un script con Yargs que permita sumar dos números desde la línea de comandos. (n1 y n2)
3. Implementa un comando que lea un archivo JSON y muestre su contenido en la terminal.
4. Modifica el script para que, si el usuario no proporciona el argumento nombre, se muestre un mensaje de error adecuado.