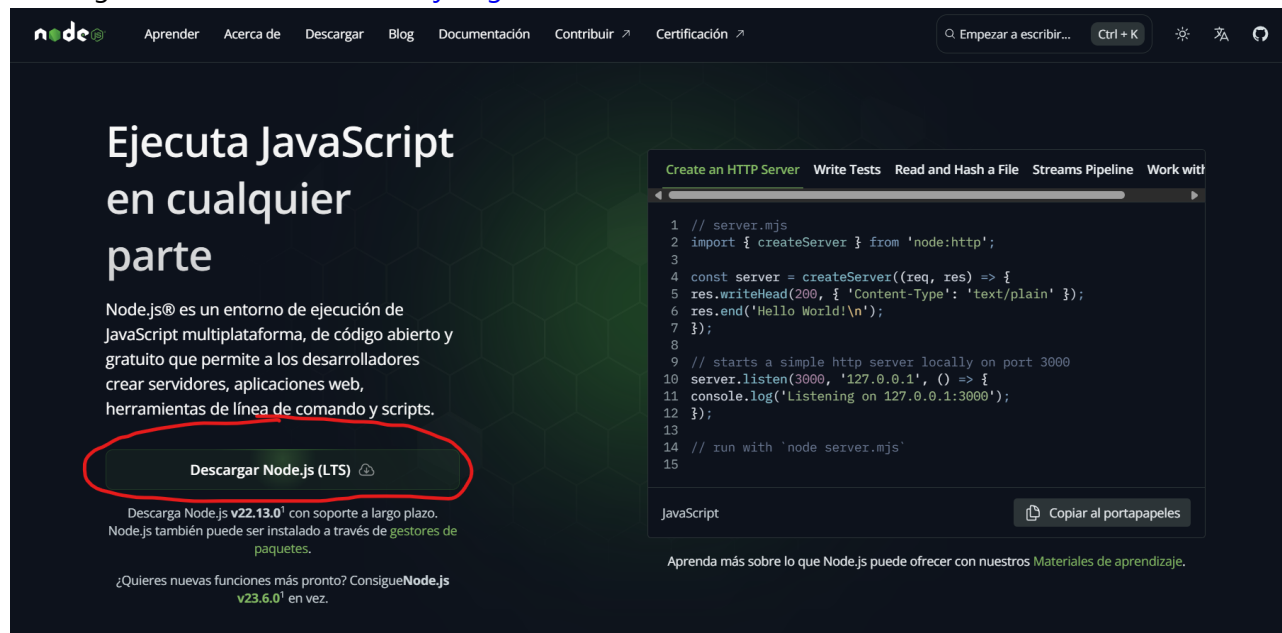


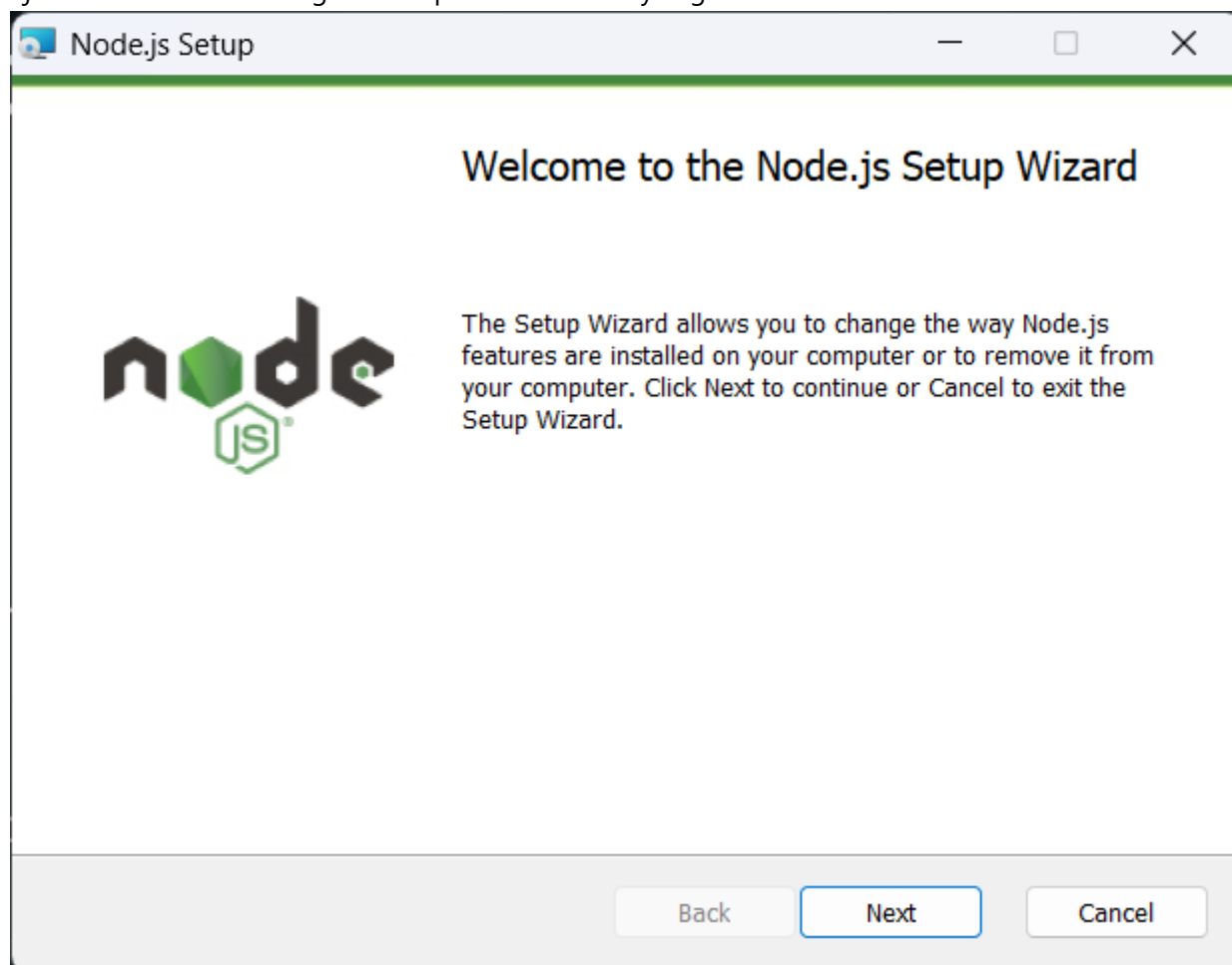
# Enunciados y Respuestas con capturas:

## 1. Instalación de Node.js en Windows 10 y Windows 11

1. Descarga el instalador desde [nodejs.org](https://nodejs.org).



2. Ejecuta el archivo descargado. Acepta los términos y elige la ruta de instalación.



3. Activa la casilla que actualiza la variable de entorno "Path" si aparece.

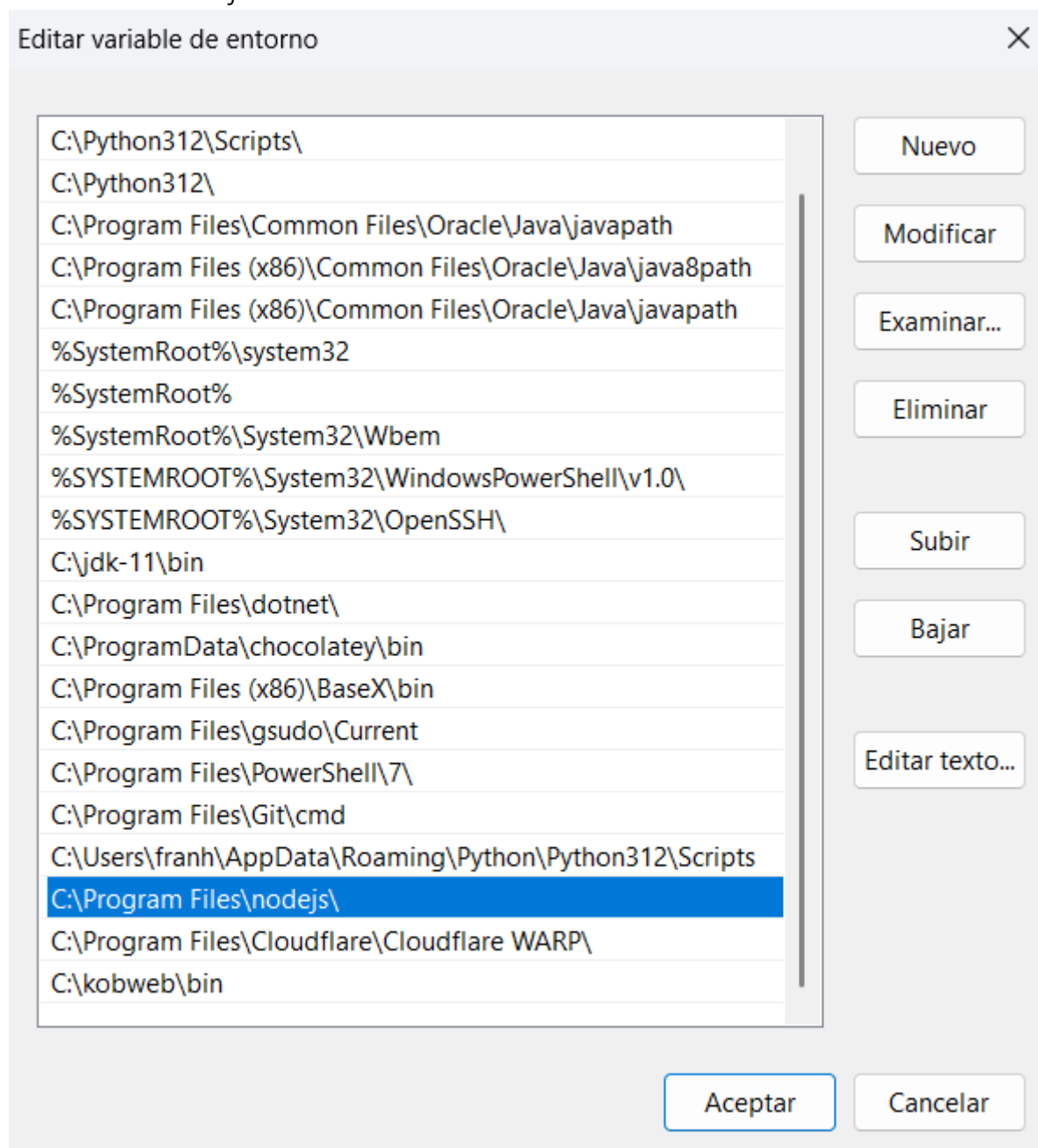
4. Finaliza la instalación y abre la terminal. Ejecuta:

```
node -v  
npm -v
```

Esto confirma la instalación correcta.

5. Si "node" no se reconoce, agrega manualmente la ruta de Node.js en "Path". En Windows 11:

- Pulsa Windows + R y escribe "SystemPropertiesAdvanced".
- Selecciona "Variables de entorno" y en "Variables del Sistema" -> "Path", añade la carpeta de instalación de Node.js.



Otra opción, en Windows 10 o Windows 11, es abrir PowerShell (Terminal en Windows 11) y ejecutar:

```
winget install --id=OpenJS.NodeJS.LTS -e
```

Node.js estará instalado y funcionando en el sistema.

2. Crea una función sencilla que imprima '¡Hola, Mundo!' en la terminal.

```
function holaMundo(){  
    console.log("¡Hola Mundo!");  
}  
holaMundo(); // Imprime ¡Hola Mundo!
```

```
[Running] - node "d:\Grado Superior\2.Segundo\AccesoADatos\UD03\NodeJS\Tarea01ConceptosBasicos\2.HolaMundo.js"  
¡Hola Mundo!  
[Done] - exited with code=0 in 0.069 seconds
```

3. Crea una función que tome dos números como argumentos e imprima su suma.

```
function sumaDosArgumentos(a, b) {  
    console.log(a + b);  
}  
// Ejemplo de uso  
sumaDosArgumentos(3, 5); // Imprime 8
```

```
[Running] - node "d:\Grado Superior\2.Segundo\AccesoADatos\UD03\NodeJS\Tarea01ConceptosBasicos\3.SumaDosArgumentos.js"  
8  
[Done] - exited with code=0 in 0.06 seconds
```

4. Crea una función que lea un archivo de texto llamado 'data.txt' y muestre su contenido en la consola.

```
var fs = require('fs');  
var path = require('path');  
  
var dir = path.join(__dirname, 'archivos_txt');  
var filePath = path.join(dir, 'data.txt');  
  
function leerArchivo() {  
    fs.readFile(filePath, 'utf8', function(err, data) {  
        if (err) {  
            return console.log(err);  
        }  
        console.log(data);  
    });  
}  
  
leerArchivo(); // Imprime el contenido del archivo data.txt
```

```
[Running] node "d:\Grado Superior\2.Segundo\AccesoADatos\UD03\NodeJS\Tarea01ConceptosBasicos\4.MostrarContenidoTXT.js"
Deserunt veniam dui nostrud do anim. Pariatur tempor ipsum pariatur in laborum cillum sit dolor occaecat culpa nisi adipisicing. Incidunt ullamco commodo officia dolor minim elit ea ut. Nisi mollit occaecat labore commodo est esse consectetur aliqua cupidatat nostrud cillum adipisicing. Dolore anim cillum sit consequat pariatur consequat nulla do eiusmod est minim. Commodo voluptate sit est velit id mollit labore et sunt laborum anim enim.
[Done] exited with code=0 in 0.058 seconds
```

5. Crea una función que escriba el texto "Hola, mundo" en un archivo llamado output.txt.

```
var fs = require('fs');
var path = require('path');

var dir = path.join(__dirname, 'archivos_txt');
var filePath = path.join(dir, 'output.txt');


if (!fs.existsSync(dir)) {
    fs.mkdirSync(dir);
}

if (!fs.existsSync(filePath)) {
    fs.writeFileSync(filePath, '');
    console.log("Archivo creado.");
}

function escribirArchivo() {
    fs.writeFile(filePath, "Hola Mundo", function(err) {
        if (err) {
            return console.log(err);
        }
        console.log("Escrito en el archivo output.txt");
    });
}

escribirArchivo(); // Imprime el contenido del archivo data.txt
```

```
[Running] node "d:\Grado Superior\2.Segundo\AccesoADatos\UD03\NodeJS\Tarea01ConceptosBasicos\5.HolaMundoOutputTXT.js"
Archivo creado.
Escrito en el archivo output.txt
[Done] exited with code=0 in 0.058 seconds
```

```
Tarea01ConceptosBasicos > archivos_txt >  output.txt
1  Hola Mundo
```

6. Crea una función que sobrescriba el archivo log.txt con el texto "Actualización completada".

```
var fs = require('fs');
var path = require('path');

var dir = path.join(__dirname, 'archivos_txt');
```

```

var filePath = path.join(dir, 'log.txt');

if (!fs.existsSync(dir)) {
  fs.mkdirSync(dir);
}

if (!fs.existsSync(filePath)) {
  fs.writeFileSync(filePath, '');
  console.log("Archivo creado.");
}

function sobrescribirArchivo() {
  fs.writeFile(filePath, "Actualización completada.", function(err) {
    if (err) {
      return console.log(err);
    }
    console.log("Sobrescribiendo en el archivo output.txt");
  });
}

/* fs.writeFile sobrescribe el contenido anterior, eliminándolo por completo.
Para agregar contenido sin borrar lo existente, hay que usar fs.appendFile. */
sobrescribirArchivo();

```

```

[Running] - node "d:\Grado Superior\2.Segundo\AccesoADatos\UD03\NodeJS\Tarea01ConceptosBasicos\6.SobrescribirLogTXT.js"
Archivo creado.
Sobrescribiendo en el archivo output.txt

[Done] - exited with code=0 in 0.063 seconds

```

```

Tarea01ConceptosBasicos > archivos_txt > log.txt
1 Actualización completada.

```

7. Crea una función que verifique si existe un archivo llamado temp.txt y, si existe, lo elimine.

```

var fs = require('fs');
var path = require('path');
var dir = path.join(__dirname, 'archivos_txt');
var filePath = path.join(dir, 'temp.txt');

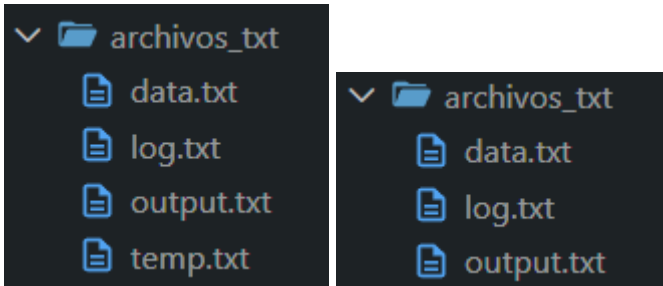
if (!fs.existsSync(dir)) {
  fs.mkdirSync(dir);
}
if (!fs.existsSync(filePath)) {
  fs.writeFileSync(filePath, '');
  console.log("Archivo creado."); /*Lo creamos antes para que exista, y así eliminarlo*/
}

function verificarEliminarTempTXT() {
  if (fs.existsSync(filePath)) {
    fs.unlinkSync(filePath);
  }
}

```

```
        console.log("Archivo eliminado.");
    } else {
        console.log("El archivo no existe.");
    }
}
verificarEliminarTempTXT();
```

```
[Running] · node · "d:\Grado·Superior\2.Segundo\AccesoADatos\UD03\NodeJS\Tarea01ConceptosBasicos\7.VerificarEliminarTempTXT.js"
Archivo creado.
Archivo eliminado.
[Done] · exited with code=0 in 0.059 seconds
```



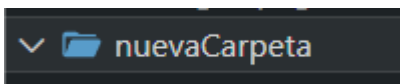
8. Crea una función que cree un directorio llamado nuevaCarpeta si este no existe.

```
var fs = require('fs');
var path = require('path');

var dir = path.join(__dirname, 'nuevaCarpeta');

function crearNuevaCarpeta() {
    if (!fs.existsSync(dir)) {
        fs.mkdirSync(dir);
        console.log("Carpeta creada.");
    } else {
        console.log("La carpeta ya existe.");
    }
}
crearNuevaCarpeta();
```

```
[Running] · node · "d:\Grado·Superior\2.Segundo\AccesoADatos\UD03\NodeJS\Tarea01ConceptosBasicos\8.CrearNuevaCarpeta.js"
Carpeta creada.
[Done] · exited with code=0 in 0.06 seconds
```



9. Crea una función que elimine un directorio llamado carpetaAntigua si este existe.

```
var fs = require('fs');
var path = require('path');

var dir = path.join(__dirname, 'carpetaAntigua');

function eliminarCarpetaAntigua() {
  if (fs.existsSync(dir)) {
    fs.rmdirSync(dir);
    console.log("Carpeta eliminada.");
  } else {
    console.log("La carpeta no existe.");
  }
}

eliminarCarpetaAntigua();
```

> carpetaAntigua

```
[Running] · node · "d:\Grado Superior\2.Segundo\AccesoADatos\UD03\NodeJS\Tarea01ConceptosBasicos\9.EliminarCarpetaAntigua.js"
Carpeta eliminada.
```

```
[Done] · exited with code=0 in 0.058 seconds
```

```
[Running] · node · "d:\Grado Superior\2.Segundo\AccesoADatos\UD03\NodeJS\Tarea01ConceptosBasicos\9.EliminarCarpetaAntigua.js"
La carpeta no existe.
```

```
[Done] · exited with code=0 in 0.053 seconds
```

10. Crea una función que lea el archivo largeData.txt en bloques y muestre cada bloque en la consola.

```
var fs = require('fs');
var path = require('path');

var dir = path.join(__dirname, 'archivos_txt');

function leerArchivoEnBloques() {
  const archivo = path.join(dir, 'largeData.txt');
  const stream = fs.createReadStream(archivo, { highWaterMark: 1024 });

  stream.on('data', (chunk) => {
    console.log(chunk.toString());
  });

  stream.on('error', (err) => {
    console.error('Error:', err);
  });

  stream.on('end', () => {
    console.log('Lectura en bloques finalizada.');
```

```
leerArchivoEnBloques();
```

```
[Running] node "d:\Grado Superior\2.Segundo\AccesoADatos\UD03\NodeJS\Tarea01ConceptosBasicos\10.LeerLargeDataTXT.js"
Elit laboris sit amet labore. Nisi sunt mollit occaecat minim quis consequat laboris exercitation dolore aliqua ad sint. Sunt cupidatat enim culpa pariatur nostrud commodo. Non aliqua dolor sit.
eiusmod enim incididunt. Ex excepteur minim qui proident est dolore veniam non esse ad ea id. Mollit fugiat id pariatur magna consequat dolor. Aliquip magna qui anim occaecat culpa occaecat amet.

Est sunt officia in esse velit ipsum eiusmod incididunt proident dolore labore duis. Ex incididunt duis consequat aliquip nisi labore. Culpa aliqua do sit enim quis officia qui.

Anim id ex magna minim culpa in deserunt cillum. Officia nisi velit cillum consequat elit cupidatat velit nostrud. Est consequat quis velit eiusmod incididunt cupidatat anim duis mollit aliquip.
Labore non nisi consequat adipisicing cupidatat nulla laborum consequat sint. Nostrud dolore ullamco ipsum voluptate et nostrud voluptate reprehenderit consectetur commodo aute ut. Ipsum laborum
reprehenderit sunt officia. Tempor cillum sint cillum offici
a dolore eiusmod aliqua aute aute ad.

Elit occaecat magna do est aute proident do dolore nostrud eu dolore. Amet quis veniam cupidatat est amet esse amet laboris occaecat officia. Ad magna adipisicing ad officia dolor eu qui non
officia voluptate nisi duis culpa non.

Quis deserunt id eiusmod elit ex duis enim aliqua. Commodo qui non sit laborum sint. Sint dolor laborum aute ullamco. Non esse cillum consectetur nostrud adipisicing. Incidunt duis ipsum qui
aliquip duis incididunt anim eu in non dolore do.

Non id occaecat nisi et et laboris officia labore non elit commodo. Sint anim aliquip tempor quis excepteur et nisi velit in anim aliqua non consequat et. Ipsum adipisicing dolor anim do ullamco.
Irure culpa nulla laboris ut exercitation aliqua. Aliquip in esse cupidatat labore cillum aliqua voluptate ea proident. Mollit amet eiusmod proident consectetur minim proident sunt mollit
consequat ut occaecat sint. Consectetur eiusmod non cillum eu.

Dolore laboris nisi incididunt qui. Non enim sint est
ullamco excepteur ea tempor mollit velit aliqua. Minim dolor velit enim proident. Elit elit ullamco consequat elit officia cupidatat ex esse anim in nulla.

Nisi nulla qui dolore esse enim cupidatat nostrud occaecat qui ex. In do et exercitation ex do laboris magna ea. Reprehenderit consectetur in reprehenderit culpa occaecat dolor tempor irure. Enim
eiusmod dolore cupidatat quis reprehenderit quis labore reprehenderit aute ex occaecat proident aliqua ea. Voluptate nisi est sunt tempor amet ad laboris do ipsum elit qui ea.

Fugiat aute nisi velit duis do tempor veniam ullamco. Velit nisi minim voluptate non sint. Consequat cupidatat fugiat adipisicing cillum ipsum aliquip ad proident nisi cillum excepteur nisi
nostrud aliqua.

Tempor voluptate voluptate occaecat ad. Est cupidatat esse sint laborum reprehenderit cillum eiusmod. Sit amet ipsum aliquip culpa anim qui non excepteur. Ex Lorem tempor aliquip id id esse
eiusmod quis magna minim.
Lectura en bloques finalizada.

[Done] exited with code=0 in 0.061 seconds
```

## 11. Crea una función que copie el contenido del archivo source.txt al archivo destination.txt.

```
var fs = require('fs');
var path = require('path');

var dir = path.join(__dirname, 'archivos_txt');

function copiarContenido() {
  const source = path.join(dir, 'source.txt');
  const destination = path.join(dir, 'destination.txt');
  fs.copyFile(source, destination, (err) => {
    if (err) {
      console.error('Error al copiar contenido:', err);
    } else {
      console.log('Contenido copiado con éxito.');
```



## 12. Crea una función que utilice pipe para leer el contenido de entrada.txt y escribirlo en salida.txt.


```
var fs = require('fs');
var path = require('path');

var dir = path.join(__dirname, 'archivos_txt');

function usarPipe() {
  const readStream = fs.createReadStream(path.join(dir, 'entrada.txt'));
  const writeStream = fs.createWriteStream(path.join(dir, 'salida.txt'));
  readStream.pipe(writeStream).on('finish', () => {
    console.log('Contenido transferido correctamente mediante pipe.');
```

```
  });
}
```


```
usarPipe();
```

Tarea01ConceptosBasicos > archivos\_txt >  entrada.txt

```
1  Il ·giudizio·finale·sta·per·essere·emesso
2  Nessuno·può·emendarsi·dal·peccato·che·scorre·nelle·vene
3  Tu·sei·senza·peccato?
4  Quanto·sarà·pesante·il·mio·castigo?
```

[Running] ·node ·"d:\Grado·Superior\2.Segundo\AccesoADatos\UD03\NodeJS\Tarea01ConceptosBasicos\12.PipeEntradaTXTSalidaTXT.js"  
Contenido·transferido·correctamente·mediante·pipe.

[Done] ·exited·with·code=0·in·0.062·seconds

Tarea01ConceptosBasicos > archivos\_txt >  salida.txt

```
1  Il ·giudizio·finale·sta·per·essere·emesso
2  Nessuno·può·emendarsi·dal·peccato·che·scorre·nelle·vene
3  Tu·sei·senza·peccato?
4  Quanto·sarà·pesante·il·mio·castigo?
```

## 13. Crea una función que verifique si existe un directorio llamado backup y lo cree si no existe.

```
var fs = require('fs');
var path = require('path');

var dir = path.join(__dirname, 'backup');


function crearNuevaCarpeta() {
  if (!fs.existsSync(dir)) {
    fs.mkdirSync(dir);
    console.log("Carpeta creada.");
  } else {
```

```
        console.log("La carpeta ya existe.");
    }
}

crearNuevaCarpeta();
```

```
[Running] · node · "d:\Grado Superior\2.Segundo\AccesoADatos\UD03\NodeJS\Tarea01ConceptosBasicos\13.VerificarCrearBackup.js"
Carpeta creada.
```

```
[Done] · exited with code=0 in 0.057 seconds
```

>  backup

```
[Running] · node · "d:\Grado Superior\2.Segundo\AccesoADatos\UD03\NodeJS\Tarea01ConceptosBasicos\13.VerificarCrearBackup.js"
La carpeta ya existe.
```

```
[Done] · exited with code=0 in 0.058 seconds
```