研修資料

情報学研究科伊藤研　　丁　世堯

2022年5月15日

1. 覚えるべき基本のLinuxコマンド

覚えるべき基本のLinuxコマンド

ls: 現在のディレクトリのファイルをリストに表示
cd: カレントフォルダを変更する（ cd ：change directory）

例：ディレクトリ”~”の下のファイル・フォルダを表示

```
(base) user@s106:~$ ls
Anaconda3-2021.11-Linux-x86_64.sh  Music       alpaca-lora-create-news-title
Desktop                            Pictures    anaconda3
Documents                          Public      cog_stanford_alpaca
Downloads                          Templates   nanoGPT
FastChat                           Videos      nanoGPTnv
```

例：カレントフォルダ”~”を、その下のフォルダ“nanoGPTnv”に変更

```
(base) user@s106:~$ cd ./nanoGPTnv
(base) user@s106:~/nanoGPTnv$ ls
LICENSE      bench.py          model.py            scaling_laws.ipynb
README.md    config            out                 train.py
__pycache__  configurator.py   sample.py           transformer_sizing.ipynb
assets       data              sample_wikiJP2.py   wandb
```

覚えるべき基本のLinuxコマンド

pwd:カレントフォルダの絶対パスを表示
cat: ファイルの内容を表示

例：カレントフォルダ“nanoGPTnv”の絶対パスを表示

```
(base) user@s106:~/nanoGPTnv$ pwd
/home/user/nanoGPTnv
```

例：README.mdファイルの内容を表示

```
(base) user@s106:~/nanoGPTnv$ cat README.md
```

実行結果

```
## troubleshooting

Note that by default this repo uses PyTorch 2.0 (i.e. `torch.compile`). This is
fairly new and experimental, and not yet available on all platforms (e.g. Window
s). If you're running into related error messages try to disable this by adding
`--compile=False` flag. This will slow down the code but at least it will run.

For some context on this repository, GPT, and language modeling it might be help
ful to watch my [Zero To Hero series](https://karpathy.ai/zero-to-hero.html). Sp
ecifically, the [GPT video](https://www.youtube.com/watch?v=kCc8FmEb1nY) is popu
lar if you have some prior language modeling context.

For more questions/discussions feel free to stop by **#nanoGPT** on Discord:

[![](https://dcbadge.vercel.app/api/server/3zy8kqD9Cp?compact=true&style=flat)](
https://discord.gg/3zy8kqD9Cp)

## acknowledgements

All nanoGPT experiments are powered by GPUs on [Lambda labs](https://lambdalabs.
com), my favorite Cloud GPU provider. Thank you Lambda labs for sponsoring nanoG
PT!
(base) user@s106:~/nanoGPTnv$
```

覚えるべき基本のLinuxコマンド

mkdir: 新しいディレクトリ（フォルダ）を作成
touch: 新しいファイルを作成

例：カレントフォルダ“nanoGPTnv”の下で、フォルダtutorial_testDicを作成

```
(base) user@s106:~/nanoGPTnv$ mkdir tutorial_testDic
(base) user@s106:~/nanoGPTnv$ ls
LICENSE       config            sample.py                tutorial_testDic
README.md     configurator.py   sample_wikiJP2.py        wandb
__pycache__   data              scaling_laws.ipynb
assets        model.py          train.py
bench.py      out               transformer_sizing.ipynb
```

例：フォルダ「tutorial_testDic」に入って、ファイルnewfile.txtを作成

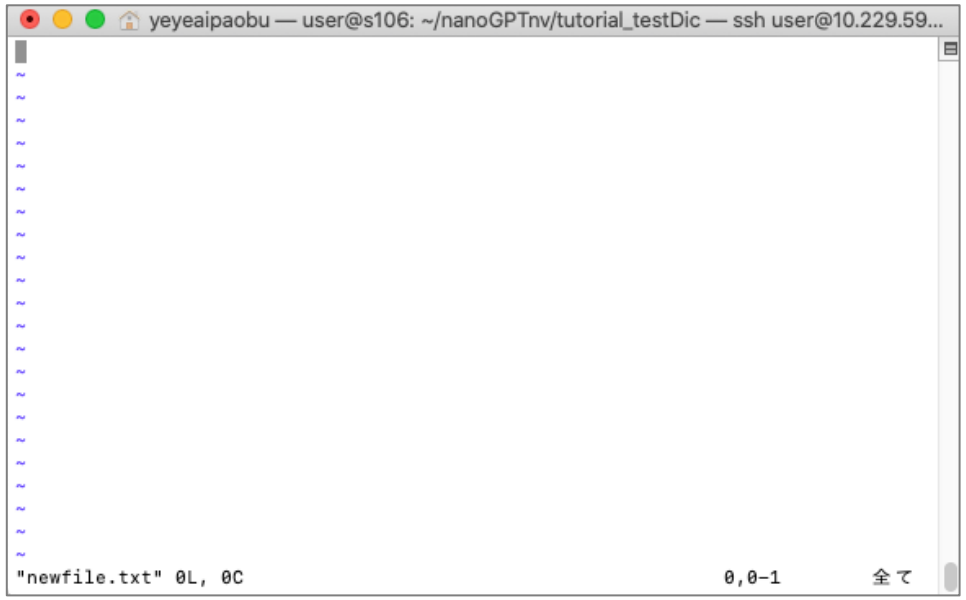
```
(base) user@s106:~/nanoGPTnv$ cd ./tutorial_testDic
(base) user@s106:~/nanoGPTnv/tutorial_testDic$ touch newfile.txt
(base) user@s106:~/nanoGPTnv/tutorial_testDic$ ls
newfile.txt
```

覚えるべき基本のLinuxコマンド

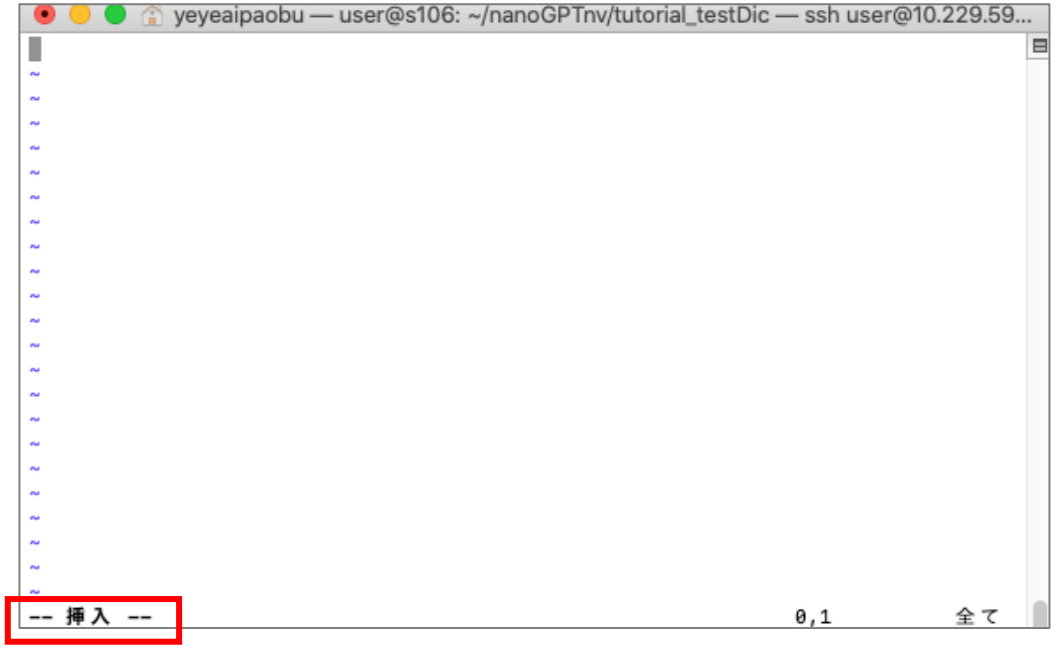
vim: ファイルの編集を行う

```
(base) user@s106:~/nanoGPTnv/tutorial_testDic$ vim newfile.txt
```

実行結果



「i」を入力して、編集モードになる

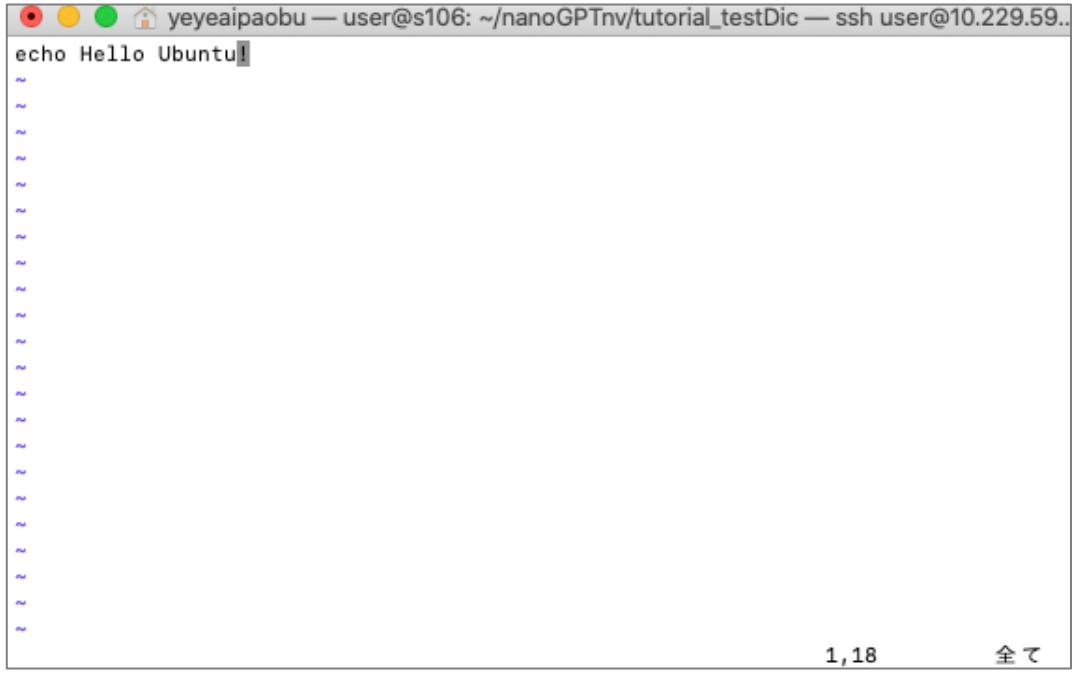


覚えるべき基本のLinuxコマンド

vim: ファイルの編集を行う

- ボタン「Esc」を押して、編集モードから退出
- 「:wq」を入力して、ボタン「Enter/return」を押して、入力された内容が保存された
- 保存せず退出する場合、「:wq」を「:q!」に変更

例：「echo Hello Ubuntu！」を入力



例：newfile.txtをbashコマンドで実行すれば、「 Hello Ubuntu！」がプリントされた

```
(base) user@s106:~/nanoGPTnv/tutorial_testDic$ bash newfile.txt
Hello Ubuntu!
```

覚えるべき基本のLinuxコマンド

rm: ファイルを削除する

例：フォルダ「tutorial_testDic」のファイルnewfile.txtを削除

```
(base) user@s106:~/nanoGPTnv/tutorial_testDic$ rm newfile.txt
(base) user@s106:~/nanoGPTnv/tutorial_testDic$ ls
(base) user@s106:~/nanoGPTnv/tutorial_testDic$
```

rm -rf: rm -rfコマンドは、ディレクトリやその中のファイルを再帰的に削除するために使用されます。
-rオプションは再帰的な削除を指定し、-fオプションは確認メッセージを表示せずに削除を実行します。

注意が必要であり、**誤った使用方法によって重要なファイルやディレクトリが削除される可能性があるため、慎重に使用する必要があります**！

例えば、rm -rf「tutorial_testDic」と入力すると、ディレクトリ「tutorial_testDic」とその中のすべてのファイルが再帰的に削除されます！！

覚えるべき基本のLinuxコマンド

課題1:
自分のラップトップで、ターミナルを立ち上げて、“Homework”という新しいディレクトリを作成する。
“Homework”内に“homework.txt”という新しいファイルを作成する。
“homework.txt”に“Hello, CREST!”と書き込んで、bashコマンドで実行する。

2. Anaconda＋専用Python仮想環境の構築

Anaconda＋専用Python仮想環境の構築

パッケージのダウンロード

- `$ pwd`
  - 現在のディレクトリが/home/ユーザ名/であることを確認
- `$ wget` https://repo.anaconda.com/archive/Anaconda3-2023.03-1-Linux-x86_64.sh
  - 各自PCへのインストールの場合、WindowsやMac用のパッケージをダウンロード
  - https://www.anaconda.com/download
- `$ ls -l`
  - /home/ユーザ名/の下に、Anaconda3-2023.03-1-Linux-x86_64.shがダウンロードされていることを確認

Anaconda＋専用Python仮想環境の構築

インストール

- $ bash Anaconda3-2023.03-1-Linux-x86_64.sh
  - 対話質問に従ってインストールする

```
Welcome to Anaconda3 2020.02

In order to continue the installation process, please review the license
agreement.
Please, press ENTER to continue
>>>  # <---------------------------------- エンターを入力
==================================
Anaconda End User License Agreement

...

Do you accept the license terms? [yes|no]
[no] >>> yes  # <---------------------------- yes を入力

Anaconda3 will now be installed into this location:
/home/user1/anaconda3

  - Press ENTER to confirm the location
  - Press CTRL-C to abort the installation
  - Or specify a different location below

[/home/user1/anaconda3] >>>  # <------------ エンターを入力
PREFIX=/home/user1/anaconda3

...
```

Anaconda＋専用Python仮想環境の構築

初期設定

- インストール完了後、一旦ログアウトしてから再度ログインする
  - ログインしたら、自動的に Conda環境 が有効となり、Anacondaが提供する python コマンドなどが使えるようになる.
    - ターミナルで(base) ユーザ名@s##:~Sを表示
  - Conda環境が自動的に有効にならないように設定する場合は、ターミナルで以下を実行する
    - $ conda config --set auto_activate_base false
    - この場合、Conda環境を利用する時には、次のコマンドでConda環境を有効化する
      - $ conda activate

- $ conda –Vで確認し、conda 23.3.1が出たらインストールが成功
  - $ which conda や$ which python, $python –Vなども確認

Anaconda＋専用Python仮想環境の構築

仮想Python環境の作成

- $ conda create -n llm python=3.8
  - 専用仮想Python環境llm(命名は任意)を作成

- $ conda activate llm
  - 専用仮想Python環境llmを有効化
  - コマンドラインの一番前の(base) -> (llm)に変化
  - llm仮想環境のディレクトリを確認
    - $ cd anaconda3/envs/
    - $ ls -l

- $ conda deactivate
  - llmを無効化し、再び(base)に戻る

Anaconda＋専用Python仮想環境の構築

作成した仮想Python環境にライブラリをインストール

- `$ conda activate llm`
  - 以降の作業は専用仮想Python環境`llm`において行う

- `/home/ユーザ名/`において`llm`をインストール
  - `$ cd /home/ユーザ名/`
  - $ git clone https://github.com/tloen/alpaca-lora.git
  - $ cd alpaca-lora/
  - $ pip install -r requirements.txt
    - エラーが出る場合、requirements.txtを修正
  - `$ pip list`
    - インストールしたライブラリー覧を確認

Anaconda＋専用Python仮想環境の構築

課題2: 自分のラップトップで、「Homework」という仮想環境（ python=3.9 ）を作って、numpy 、matplotlibやnetworkxのライブラリを、「Homework」にインストールする。以下のコードを一つの.pyファイルに保存して、 Activiteした「Homework」環境でコマンドpythonで実行する。

```
import numpy as np
import networkx as nx
import matplotlib.pyplot as plt

# Creating a NumPy array
arr = np.array([1, 2, 3, 4, 5])
print("NumPy array:", arr)

# Creating a graph using NetworkX
G = nx.Graph()
G.add_edge(1, 2)
G.add_edge(2, 3)
G.add_edge(3, 4)
G.add_edge(4, 5)
print("NetworkX graph nodes:", G.nodes())
print("NetworkX graph edges:", G.edges())

# Plotting the graph using Matplotlib
nx.draw(G, with_labels=True)
plt.show()
```

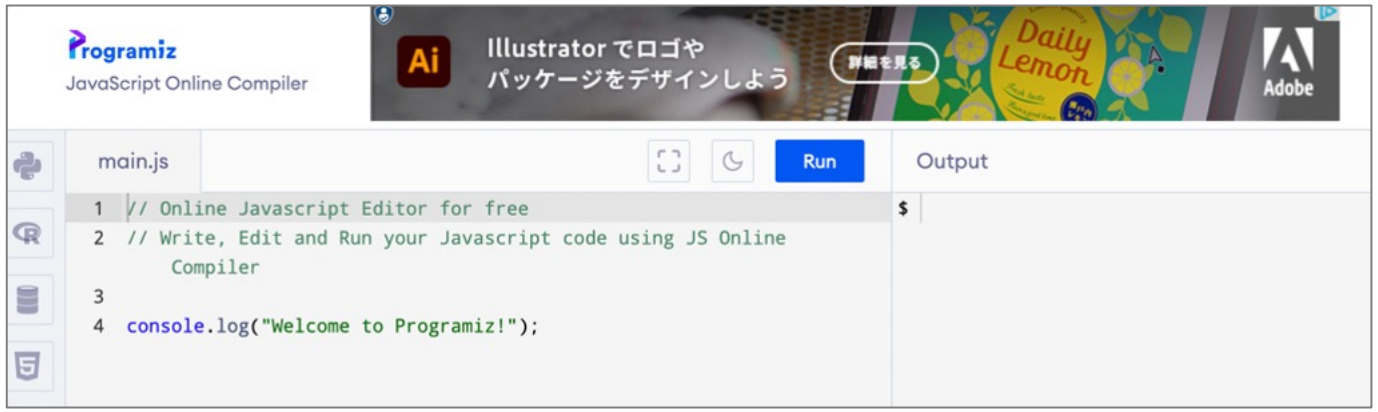
3. JavaScriptの基礎知識

JavaScriptの基礎知識

- JavaScript（ジャバスクリプト）は、ウェブ開発やアプリケーション開発に広く使用されるプログラミング言語です。
- JavaScriptは動的なコンテンツやインタラクティブな機能をウェブページに追加するために使用されます。
- さらに、サーバーサイドの開発やモバイルアプリ開発にも使用されます。

- Programizは簡単に使えるJavaScript Online Compilerです。

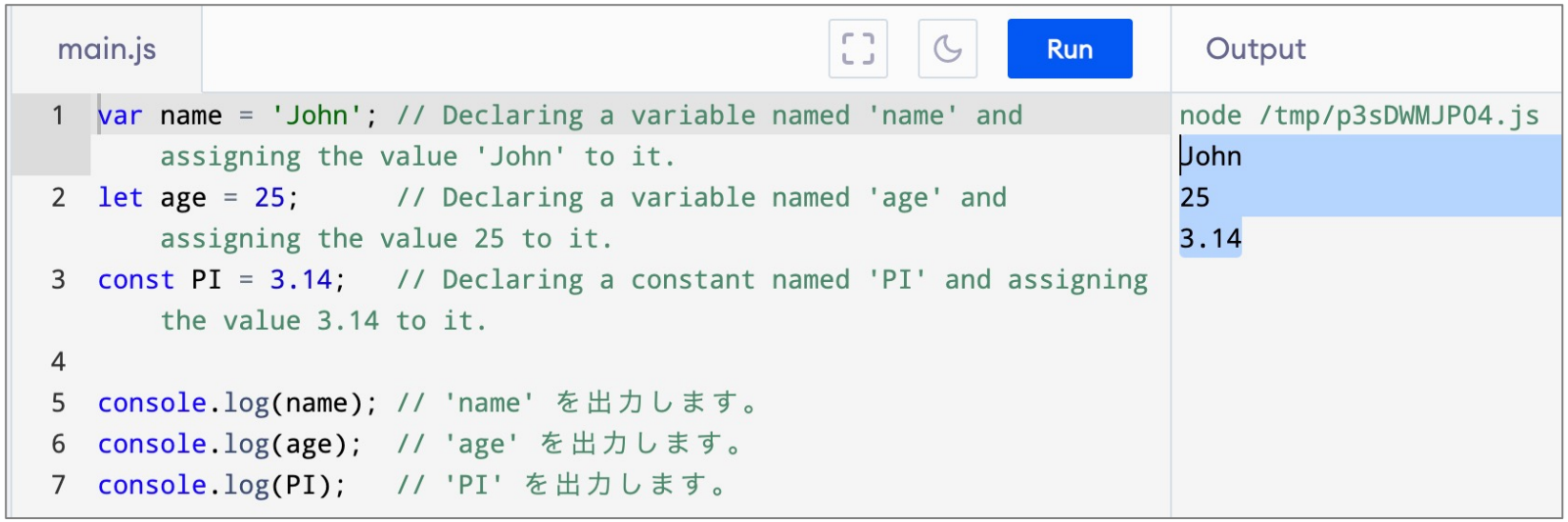
https://www.programiz.com/javascript/online-compiler/



JavaScriptの基礎知識

変数とデータ型： JavaScriptでは、var、let、またはconstキーワードを使用して変数を宣言できる。JavaScriptには、文字列、数値、ブール値、配列、オブジェクトなど、さまざまなデータ型がある。

```
var name = 'John'; // Declaring a variable named 'name' and assigning the value 'John' to it.
let age = 25; // Declaring a variable named 'age' and assigning the value 25 to it.
const PI = 3.14; // Declaring a constant named 'PI' and assigning the value 3.14 to it.

console.log(name); // 'name' を出力します。
console.log(age); // 'age' を出力します。
console.log(PI); // 'PI' を出力します。
```



JavaScriptの基礎知識

関数：関数を使用すると、特定のタスクを実行するコードをグループ化できる。

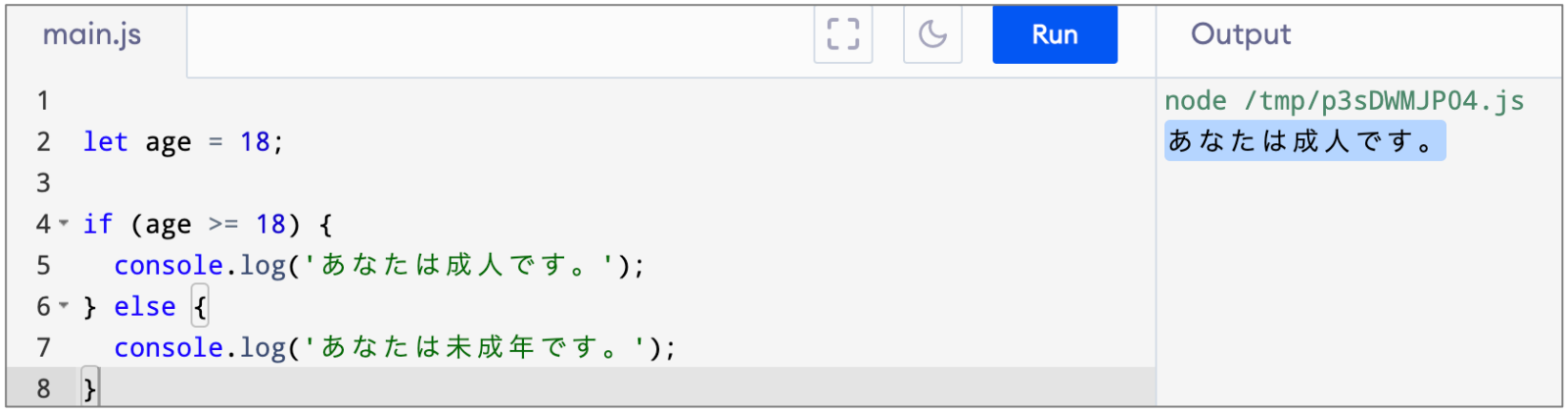
functionキーワードを使用して関数を定義できる。

```
function greet(name) {
        console.log('こんにちは、' + name + 'さん！');
}

greet('John'); // 引数 'John' を使って 'greet' 関数を呼び出します。
```

JavaScriptの基礎知識

条件文：条件文を使用すると、特定の条件に基づいて異なるコードブロックを実行できる。
JavaScriptでは、if、else if、else文を提供している。

```
let age = 18;

if (age >= 18) {
  console.log('あなたは成人です。');
} else {
  console.log('あなたは未成年です。');
}
```

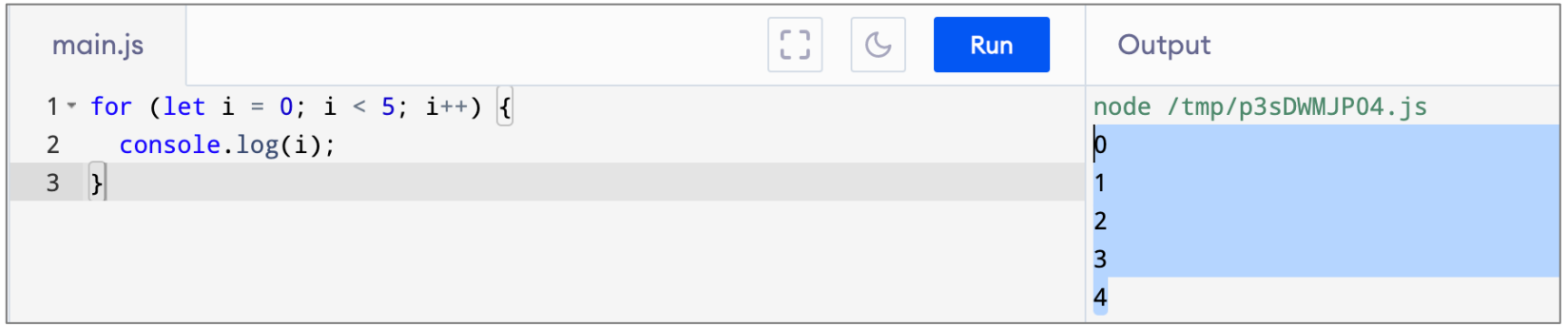


JavaScriptの基礎知識

ループ：ループは、コードブロックを繰り返し実行するために使用される。

JavaScriptでは、forループとwhileループをサポートしている。

```
for (let i = 0; i < 5; i++) {
  console.log(i);
}
```

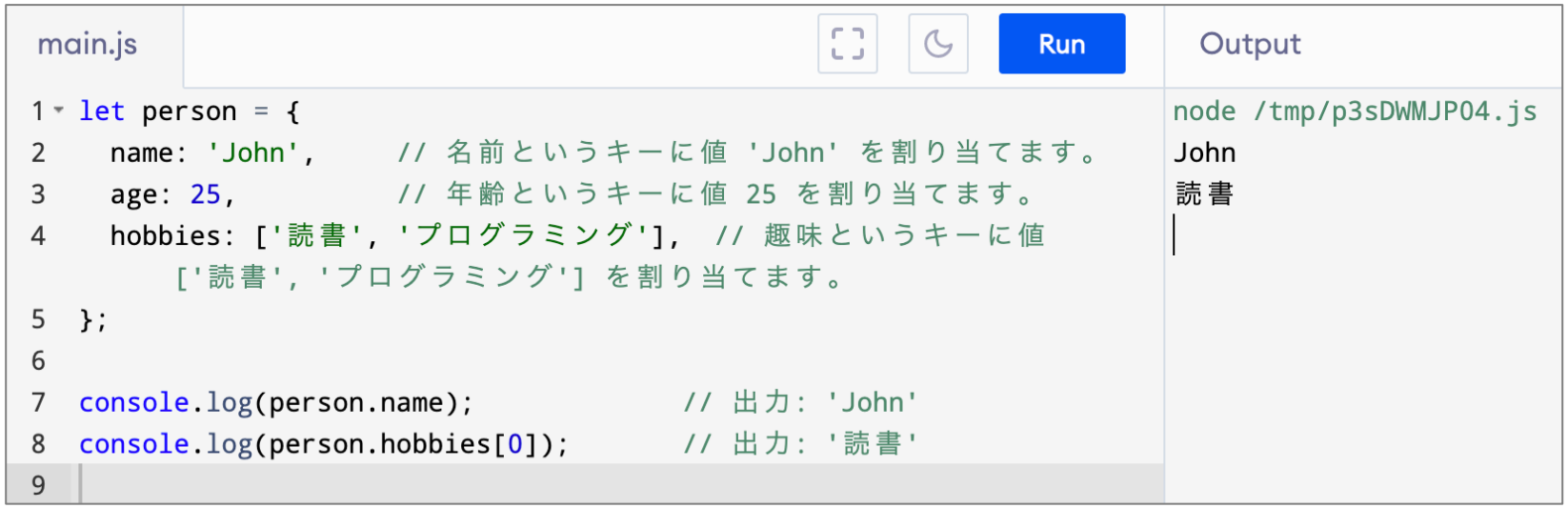


JavaScriptの基礎知識

オブジェクト:オブジェクトを使用すると、キーと値のペアを格納することができる。

各値は対応するキーを使用してアクセスされる。

```
let person = {
  name: 'John',    // 名前というキーに値 'John' を割り当てます。
  age: 25,         // 年齢というキーに値 25 を割り当てます。
  hobbies: ['読書', 'プログラミング'],  // 趣味というキーに値 ['読書', 'プログラミング'] を割り当てます。
};
console.log(person.name);           // 出力: 'John'
console.log(person.hobbies[0]);     // 出力: '読書'
```



JavaScriptの基礎知識

課題3: Programizにて、以下のコードを実行する

1. `students`という配列を作成し、少なくとも5人の学生の名前を要素として持つように。

2. `grades`という配列を作成し、対応する学生の成績を要素として持つようにする。学生の名前と成績の順序は対応している必要がある。

3. `reportCard`というオブジェクトを作成し、学生の名前と対応する成績をキーと値のペアとして持つようにする。

4. `calculateAverage`という関数を作成し、`grades`配列の平均成績を計算して返す。

5. `highestGrade`という関数を作成し、最高成績とその学生の名前を返す。