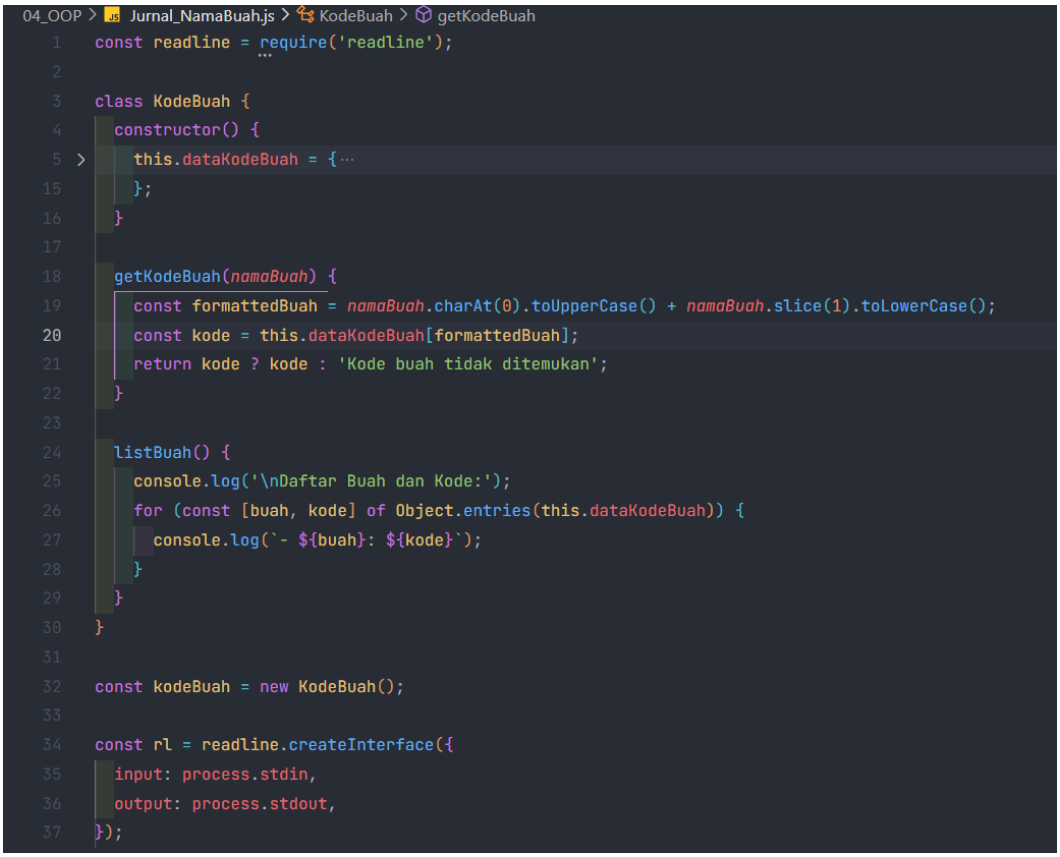
Jurnal Modul 4

1. Jurnal_NamaBuah.js

Code :



```
04_OOP > Jurnal_NamaBuah.js > KodeBuah > getKodeBuah
1   const readline = require('readline');
2
3   class KodeBuah {
4     constructor() {
5       this.dataKodeBuah = {···
15      };
16    }
17
18    getKodeBuah(namaBuah) {
19      const formattedBuah = namaBuah.charAt(0).toUpperCase() + namaBuah.slice(1).toLowerCase();
20      const kode = this.dataKodeBuah[formattedBuah];
21      return kode ? kode : 'Kode buah tidak ditemukan';
22    }
23
24    listBuah() {
25      console.log('\nDaftar Buah dan Kode:');
26      for (const [buah, kode] of Object.entries(this.dataKodeBuah)) {
27        console.log(`- ${buah}: ${kode}`);
28      }
29    }
30  }
31
32  const kodeBuah = new KodeBuah();
33
34  const rl = readline.createInterface({
35    input: process.stdin,
36    output: process.stdout,
37  });
```

Di dalam Class KodeBuah terdapat data nama buah dan kode buah dalam bentuk objek. Memiliki metode getKodeBuah untuk mencari kode buah berdasarkan nama buah. metode listBuah untuk mencetak daftar buah yang tersedia. Menggunakan modul readline untuk membaca input pengguna dari terminal. Menampilkan daftar buah, lalu meminta pengguna memasukkan nama buah. Jika nama buah ditemukan, kode buah ditampilkan. Jika tidak, muncul pesan "Kode buah tidak ditemukan.".

Output :

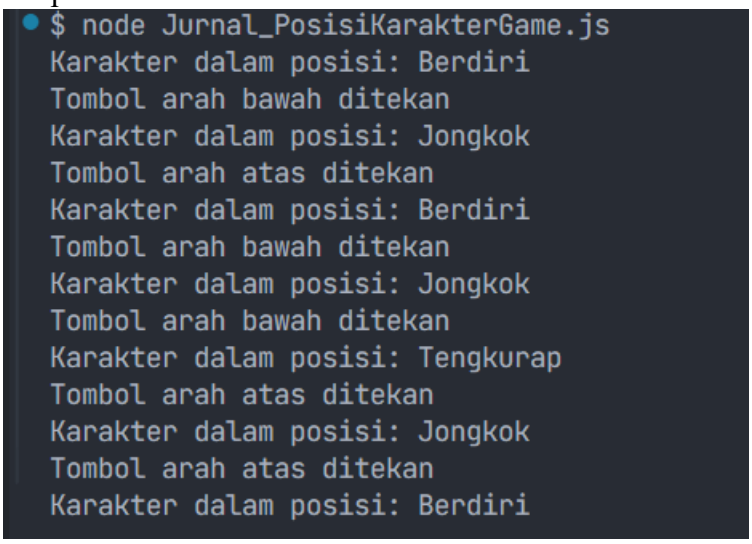
```
$ node Jurnal_NamaBuah.js

Daftar Buah dan Kode:
- Apel: A00
- Aprikot: B00
- Alpukat: C00
- Pisang: D00
- Paprika: E00
- Blackberry: F00
- Ceri: H00
- Kelapa: I00
- Jagung: J00

Masukkan nama buah untuk mencari kode: ceri
Kode Buah untuk ceri: H00
```

2. Jurnal_PosisiKarakterGame.js

```
04_OOP > Jurnal_PosisiKarakterGame.js > ...
1   class PosisiKarakterGame {
2     constructor(NIM) {
3       this.state = 'Berdiri';
4       this.NIM = NIM;
5       this.modulus = NIM % 3;
6       this.showState();
7     }
8
9     changeState(action) {
10      const transitions = {
11        Berdiri: { TombolW: 'Terbang', TombolS: 'Jongkok' },
12        Jongkok: { TombolW: 'Berdiri', TombolS: 'Tengkurap' },
13        Tengkurap: { TombolW: 'Jongkok' },
14        Terbang: { TombolS: 'Berdiri' },
15      };
16
17      const nextState = transitions[this.state]?.[action];
18      if (nextState) {
19        this.handleSpecialOutput(this.state, nextState, action);
20        this.state = nextState;
21        this.showState();
22      } else {
23        console.log('Aksi tidak valid');
24      }
25    }
26
```

Kode di atas mendefinisikan class PosisiKarakterGame yang mengatur pergerakan karakter dalam game menggunakan teknik state-based construction. State awalnya adalah "Berdiri" dan dapat berubah ke "Jongkok", "Tengkurap", atau "Terbang" sesuai tombol yang ditekan (TombolW, TombolS, atau TombolX). Transisi antar state diatur dalam tabel transisi, dan tambahan aturan diberikan berdasarkan hasil NIM % 3, seperti menampilkan pesan khusus saat karakter berubah posisi. Implementasi ini kemudian disimulasikan dalam metode main untuk memastikan semua transisi dan output berjalan sesuai spesifikasi.

Output :


$ node Jurnal_PosisiKarakterGame.js
Karakter dalam posisi: Berdiri
Tombol arah bawah ditekan
Karakter dalam posisi: Jongkok
Tombol arah atas ditekan
Karakter dalam posisi: Berdiri
Tombol arah bawah ditekan
Karakter dalam posisi: Jongkok
Tombol arah bawah ditekan
Karakter dalam posisi: Tengkurap
Tombol arah atas ditekan
Karakter dalam posisi: Jongkok
Tombol arah atas ditekan
Karakter dalam posisi: Berdiri