1. Classes e Objetos

1.1 Classe Pessoa

```java
public abstract class Pessoa {
    public String nome;
    private String cpf;
    private int idade;

    Pessoa(String nome, String cpf, int idade){
        this.nome = nome;
        this.cpf = cpf;
        this.idade = idade;
    }


    public String getNome(){
        return nome;
    }

    public void setNome(String nome){
        this.nome = nome;
    }

    public String getCpf(){
        return this.cpf;
    }

    public void setCpf(String cpf){
        this.cpf = cpf;
    }

    public int getIdade(){
        return this.idade;
    }

    public void setIdade(int idade){
        this.idade = idade;
    }

    public void apresentar(){
        System.out.println(
            "Nome: " + this.nome +
            " | CPF: " + this.cpf +
            " | Idade: " + this.idade
        );
    }
}
```

A classe ***Pessoa*** foi definida como uma classe abstrata, funcionando como um molde para as demais classes que herdam suas características.

Por se tratar de uma classe abstrata, ela não pode ser instanciada diretamente. Portanto, foi implementado um **método construtor** responsável pela inicialização dos atributos da classe sempre que houver a criação de um objeto derivado.

```java
public abstract class Pessoa {
    public String nome;
    private String cpf;
    private int idade;

    Pessoa(String nome, String cpf, int idade){
        this.nome = nome;
        this.cpf = cpf;
        this.idade = idade;
    }

```

```java

    public String getNome(){
        return nome;
    }

    public void setNome(String nome){
        this.nome = nome;
    }

    public String getCpf(){
        return cpf;
    }

    public void setCpf(String cpf){
        this.cpf = cpf;
    }

    public int getIdade(){
        return idade;
    }

    public void setIdade(int idade){
        this.idade = idade;
    }

```

Além disso, foram definidos métodos de acesso (***getters* e *setters***) para garantir o encapsulamento dos atributos, permitindo sua leitura e modificação de forma controlada.

```
36
37      public void apresentar(){
38          System.out.println(
39              "Nome: " + this.nome +
40              " | CPF: " + this.cpf +
41              " | Idade: " + this.idade
42          );
43      }
44  }
```

Também foi implementado um método que exibe no terminal as informações contidas na classe, facilitando a visualização dos dados armazenados.

1.2 Classe Disciplina

```
1   public class Disciplina {
2       String codigo;
3       String nome;
4       int semestre;
5
6       Disciplina(String codigo, String nome, int semestre){
7           this.codigo = codigo;
8           this.nome = nome;
9           this.semestre = semestre;
10      }
11
12      @Override
13      public String toString(){
14          return  this.nome + " (Código: " + this.codigo + ")";
15      }
16  }
17
```

A classe ***Disciplina*** possui um método construtor semelhante ao da classe Pessoa, responsável por inicializar os dados no momento da criação de uma nova disciplina.

```
11
12      @Override
13      public String toString(){
14          return  this.nome + " (Código: " + this.codigo + ")";
15      }
16  }
17
```

Foi implementado também o método ***toString()***, com o objetivo de retornar uma **representação textual** do objeto. Dessa forma, ao imprimir ou exibir objetos do tipo Disciplina no terminal, serão mostrados os valores definidos em seus atributos, e não o endereço de memória em que o objeto está armazenado.