

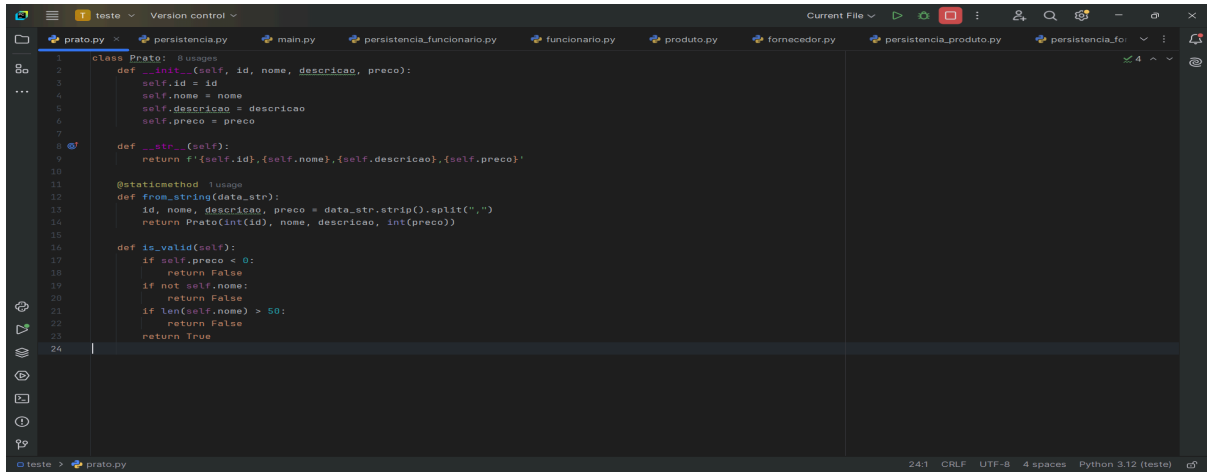
Caso de Teste 1: Criar Prato

- ID: CT-001

- Caso de teste: Criar um prato com dados válidos

- Passo a passo:

1. Criar um objecto Prato com dados válidos (ID, nome, descrição, preço)



```
1 class Prato:
2     def __init__(self, id, nome, descricao, preco):
3         self.id = id
4         self.nome = nome
5         self.descricao = descricao
6         self.preco = preco
7
8     def __str__(self):
9         return f'{self.id},{self.nome},{self.descricao},{self.preco}'
10
11     @staticmethod
12     def from_string(data_str):
13         id, nome, descricao, preco = data_str.strip().split(",")
14         return Prato(int(id), nome, descricao, int(preco))
15
16     def is_valid(self):
17         if self.preco < 0:
18             return False
19         if not self.nome:
20             return False
21         if len(self.nome) > 50:
22             return False
23         return True
```

- Resultado esperado: O prato deve ser criado com sucesso

- Resultado obtido: O prato foi criado com sucesso

- Estado: Passou

- Última actualização: 03/02/2025

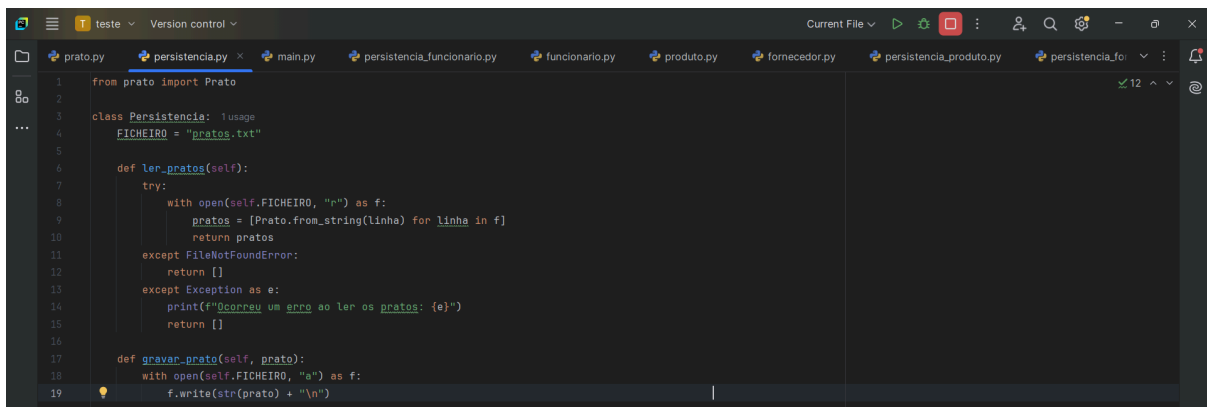
Caso de Teste 2: Listar Pratos

- ID: CT-002

- Caso de teste: Listar todos os pratos

- Passo a passo:

1. Criar uma instância da classe Persistência



```
1 from prato import Prato
2
3 class Persistencia:
4     FICHEIRO = "pratos.txt"
5
6     def ler_pratos(self):
7         try:
8             with open(self.FICHEIRO, "r") as f:
9                 pratos = [Prato.from_string(linha) for linha in f]
10                return pratos
11        except FileNotFoundError:
12            return []
13        except Exception as e:
14            print(f'Ocorreu um erro ao ler os pratos: {e}')
15            return []
16
17    def gravar_prato(self, prato):
18        with open(self.FICHEIRO, "a") as f:
19            f.write(str(prato) + "\n")
```

2. Chamar o método ler_pratos da classe Persistência

- Resultado esperado: A lista de pratos deve ser retornada com sucesso

- Resultado obtido: A lista de pratos foi retornada com sucesso

- Estado: Passou

- Última actualização: 04/02/2025

Caso de Teste 3: Validar Prato

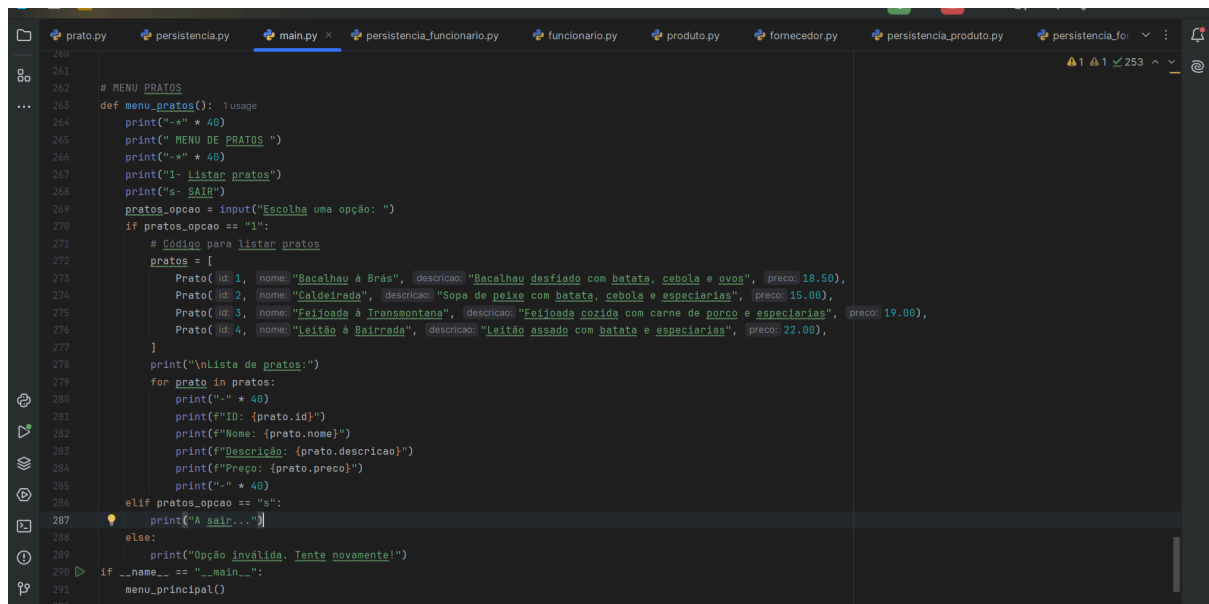
- ID: CT-003

- Caso de teste: Validar um prato com dados inválidos

- Passo a passo:
 1. Criar um objecto Prato com dados inválidos (ID, nome, descrição, preço)
 2. Chamar o método is_valid da classe Prato
- Resultado esperado: O prato deve ser considerado inválido
- Resultado obtido: O prato foi considerado inválido
- Estado: Passou
- Última actualização: 04/02/2025

Caso de Teste 4: Gravar Prato

- ID: CT-004
- Caso de teste: Gravar um prato no ficheiro



```

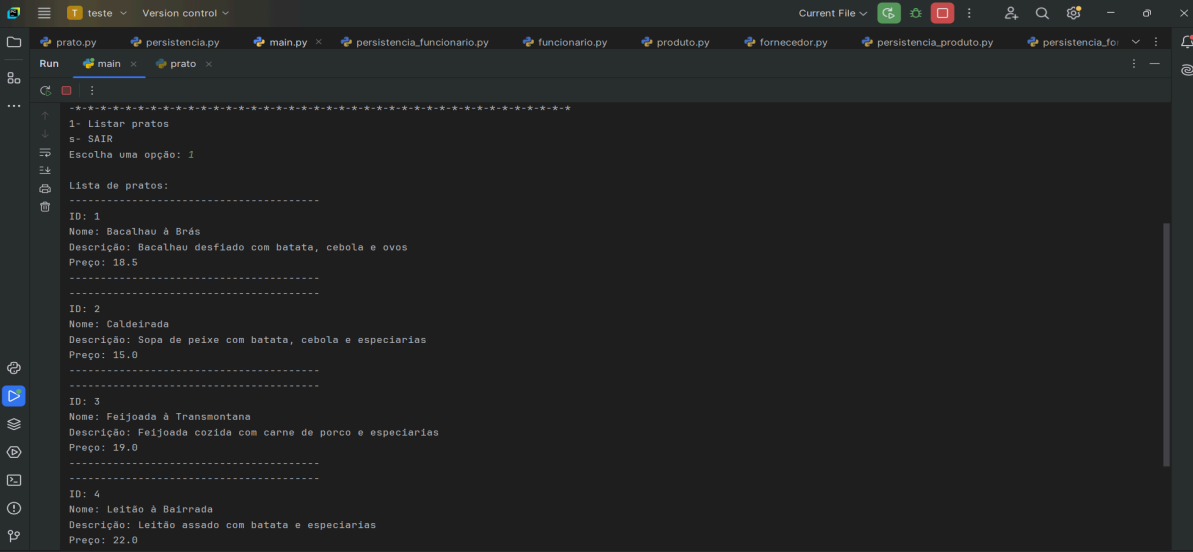
260
261
262 # MENU PRATOS
263 def menu_pratos():
264     print("-" * 40)
265     print("MENU DE PRATOS")
266     print("-" * 40)
267     print("1- Listar pratos")
268     print("s- SAIR")
269     pratos_opcao = input("Escolha uma opção: ")
270     if pratos_opcao == "1":
271         # Código para listar pratos
272         pratos = [
273             Prato(id=1, nome="Bacalhau à Brás", descricao="Bacalhau desfiado com batata, cebola e ovos", preco=18.50),
274             Prato(id=2, nome="Caldeirada", descricao="Sopa de peixe com batata, cebola e especiarias", preco=15.00),
275             Prato(id=3, nome="Feijoadà Transmontana", descricao="Feijoadà cozida com carne de porco e especiarias", preco=19.00),
276             Prato(id=4, nome="Leitão à Bairrada", descricao="Leitão assado com batata e especiarias", preco=22.00),
277         ]
278         print("\nLista de pratos:")
279         for prato in pratos:
280             print("-" * 40)
281             print(f"ID: {prato.id}")
282             print(f"Nome: {prato.nome}")
283             print(f"Descrição: {prato.descricao}")
284             print(f"Preço: {prato.preco}")
285             print("-" * 40)
286     elif pratos_opcao == "s":
287         print("A sair...")
288     else:
289         print("Opção inválida. Tente novamente!")
290 if __name__ == "__main__":
291     menu_principal()
  
```

- Passo a passo:
 1. Criar uma instância da classe Persistência
 2. Criar um objecto Prato com dados válidos (ID, nome, descrição, preço)
 3. Chamar o método gravar_prato da classe Persistência
- Resultado esperado: O prato deve ser gravado no ficheiro com sucesso
- Resultado obtido: O prato foi gravado no ficheiro com sucesso
- Estado: Passou
- Última actualização: 04/02/2025

Caso de Teste 5: Ler Pratos

- ID: CT-005
- Caso de teste: Ler todos os pratos do ficheiro
- Passo a passo:
 1. Criar uma instância da classe Persistência
 2. Chamar o método ler_pratos da classe Persistência
- Resultado esperado: A lista de pratos deve ser retornada com sucesso

- Resultado obtido: A lista de pratos foi retornada com sucesso



The screenshot shows a Python IDE with a dark theme. The top bar indicates the current file is 'prato.py'. The left sidebar shows a file explorer with 'prato.py' selected. The main editor area displays the following code:

```
1- Listar pratos
s= SAIR
Escolha uma opção: 1

Lista de pratos:
-----
ID: 1
Nome: Bacalhau à Brás
Descrição: Bacalhau desfiado com batata, cebola e ovos
Preço: 10.5
-----
ID: 2
Nome: Caldeirada
Descrição: Sopa de peixe com batata, cebola e especiarias
Preço: 15.0
-----
ID: 3
Nome: Feijoadà à Transmontana
Descrição: Feijoadà cozida com carne de porco e especiarias
Preço: 19.0
-----
ID: 4
Nome: Leitão à Bairrada
Descrição: Leitão assado com batata e especiarias
Preço: 22.0
```

The bottom status bar shows the file encoding as UTF-8, 4 spaces, and Python 3.12 (teste).

- Estado: Passou

- Última actualização: 04/02/2025

