DDP 1 - Lab 1

PaintMan

Selamat datang di Dasar-Dasar Pemrograman 1 (DDP 1)!

Anda akan berperan sebagai PaintMan, yang bertugas mengecat dinding-dinding rumah. PaintMan, yang telah mengambil MOOC DDP 1, ingin memanfaatkan kemampuan pemrogramannya untuk menunjang mata pencahariannya.

Tugas Anda pada lab kali ini adalah sebagai berikut:

- Anda membuat program untuk menghitung luas total area pengecatan pada rumah yang terdiri dari beberapa dinding.
- Program yang Anda buat akan bertanya jumlah dinding, serta panjang dan lebar masing-masing dinding.
- Setelah itu, program akan memproses input tersebut dan menginfokan luas total area pengecatan.

Contoh interaksi 1:

```
Selamat datang di PaintMan v1.0

Jumlah dinding: 3
~~
Dinding 1
Panjang (m): 3
Lebar (m): 2
Luas (m2): 6.00
~~
Dinding 2
Panjang (m): 2
Lebar (m): 1.5
Luas (m2): 3.00
~~
Dinding 3
Panjang (m): 4
Lebar (m): 2.5
Luas (m2): 10.00
~~~~
Luas total dari semua dinding adalah 19.00 m2
```

Contoh interaksi 2:

```
Selamat datang di PaintMan v1.0

Jumlah dinding: 2
~~
Dinding 1
Panjang (m): 1.5
Lebar (m): 1.5
Luas (m2): 2.25
~~
Dinding 2
Panjang (m): 1.5
Lebar (m): 3
Luas (m2): 4.50
~~~~
Luas total dari semua dinding adalah 6.75 m2
```

Anda **wajib** melengkapi template berikut dalam pengerjaan lab.

```
# lengkapi kode
print()

jml_dinding = # lengkapi kode

luas_total = 0

for i in range(jml_dinding):
    print("~~")
    print(f"Dinding {i+1}")
    p = float(input("Panjang (m): "))
    l = float(input("Lebar (m): "))
    luas = # lengkapi kode
    print(f"Luas (m2): {luas:.2f}")
    luas_total = # lengkapi kode

print("~~~~")
# lengkapi kode
```

Deliverables

Anda **dilarang** keras melakukan plagiarisme! Tidak boleh memakai ChatGPT dan sejenisnya. Kumpulkan berkas .py yang berisi kode solusi Anda dengan format penamaan: `MOOCUI2023_DDP1_[NamaLengkap]_Lab01.py`

Contoh (jangan lupa ganti nama lengkapnya untuk submisi Anda):
`MOOCUI2023_DDP1_FarizDarari_Lab01.py`

Acknowledgements

- FD
- flaticon

DDP 1 - Lab 2

Cashier Man

Selamat datang di Dasar-Dasar Pemrograman 1 (DDP 1)!

CodeBrew Café saat ini masih mencatat pembayaran pelanggan secara manual dengan menulisnya di nota kertas. Namun, seiring dengan meningkatnya jumlah pelanggan, kasir di kafe tersebut merasa kesulitan untuk mengikuti proses manual ini.

Dalam kesempatan ini, Anda akan membantu seorang kasir di CodeBrew Café untuk menyederhanakan proses pembuatan tagihan pembayaran pelanggan agar lebih mudah. Anda akan membuat sebuah program yang dapat dengan cepat dan rapi membuat file tagihan pembayaran.

Tugas Anda pada lab kali ini adalah sebagai berikut:

- Anda membuat program untuk menghitung total biaya yang perlu dibayar oleh pelanggan
- Program yang Anda buat akan bertanya nama customer dan informasi produk berupa nama, harga satuan, dan jumlah.
- Program terus meminta input informasi produk dan akan berhenti apabila Anda memasukkan string kosong sebagai input pada nama produk.
- Setelah itu, program akan memproses input tersebut dengan menginfokan total pembayaran serta nama dari file text bill.
- Program akan membuat text file yang merupakan bill pembayaran dengan nama bill yang diinfokan oleh program.
- Nama text file bill dipersonalisasikan sesuai dengan customer yang memesan.

Contoh interaksi:

```
Customer Name: alina
Product (enter to finish): iced matcha latte
Quantity: 1
Price: 23000
Product (enter to finish): hot latte
Quantity: 1
Price: 20000
Product (enter to finish): croissant
Quantity: 2
Price: 14500
Product (enter to finish):

Total Payment: Rp72,000.00

Billing data has been saved to bill_alina.txt
```

Contoh hasil text file bernama “bill_alina.txt” :

```
==========================================================================
                          CODEBREW CAFE BILL
==========================================================================
   1  |  iced matcha latte          |  Rp 23,000.00  |  Rp      23,000.00

   1  |  hot latte                  |  Rp 20,000.00  |  Rp      20,000.00

   2  |  croissant                  |  Rp 14,500.00  |  Rp      29,000.00

==========================================================================

Total Payment: Rp72,000.00
```

Anda **wajib** melengkapi template berikut dalam pengerjaan lab.

```
total_payment = 0
customer_name = input(f"Customer Name: ")

billname = #lengkapi kode
bill =  #lengkapi kode

print("=" * 74, file=bill)
print(f"{'CODEBREW CAFE BILL':^74}", file=bill)
print("=" * 74, file=bill)

while True:
    product_name = input(f"Product (enter to finish): ")

    if product_name == "":
        break

    quantity =  #lengkapi kode
    price =  #lengkapi kode
    total_price =  #lengkapi kode

    total_payment +=  #lengkapi kode
```

```
    print(f"{quantity:4d}  |  {product_name:25s} |  Rp{price:10.2f}  |
Rp{total_price:15.2f}\n", file=bill)

print("=" * 74, file=bill)
print(f"\nTotal Payment: Rp{total_payment:,.2f}\n", #lengkapi kode)
print(f"\nTotal Payment: Rp{total_payment:,.2f}\n")

 #lengkapi kode
print(f"Billing data has been saved to {billname}")
```

Deliverables

Anda **dilarang** keras melakukan plagiarisme! Tidak boleh memakai ChatGPT dan sejenisnya. Kumpulkan berkas .py yang berisi kode solusi Anda dengan format penamaan: `MOOCUI2023_DDP1_[NamaLengkap]_Lab02.py`

Contoh (jangan lupa ganti nama lengkapnya untuk submisi Anda):
`MOOCUI2023_DDP1_FarizDarari_Lab02.py`

Acknowledgements

- FD
- AD
- flaticon

DDP 1 - Lab 3

ThunderMan

Selamat datang di Dasar-Dasar Pemrograman 1 (DDP 1)!

Selamat datang di Kota Codeburg, pusat teknologi yang maju di mana aliran kode dan inovasi terus mengalir. Namun, ketenangan kota ini terganggu oleh serangan tiba-tiba dari seorang penjahat sengit yang dikenal sebagai EvilGuy. Kota ini dalam bahaya!

Di saat genting ini, terbitlah sosok pahlawan legendaris yang melambangkan keberanian dan keadilan: ThunderMan! Bersenjatakan kekuatan petir yang luar biasa, ThunderMan bangkit sebagai harapan satu-satunya untuk melawan kegelapan yang dibawa oleh EvilGuy. Kini, takdir Kota Codeburg bergantung pada perjuangan antara ThunderMan dan EvilGuy. Pertarungan sengit antara kebaikan dan kejahatan yang akan menentukan masa depan kota ini.

Kamu memiliki peran penting dalam menyelamatkan Codeburg. Bimbinglah ThunderMan melalui kota yang terancam ini dan hadapilah setiap serangan jahat yang dilancarkan oleh EvilGuy. Dengan pilihanmu yang bijaksana, bisakah ThunderMan menyelamatkan Codeburg dari ancaman kehancuran yang mengintai?

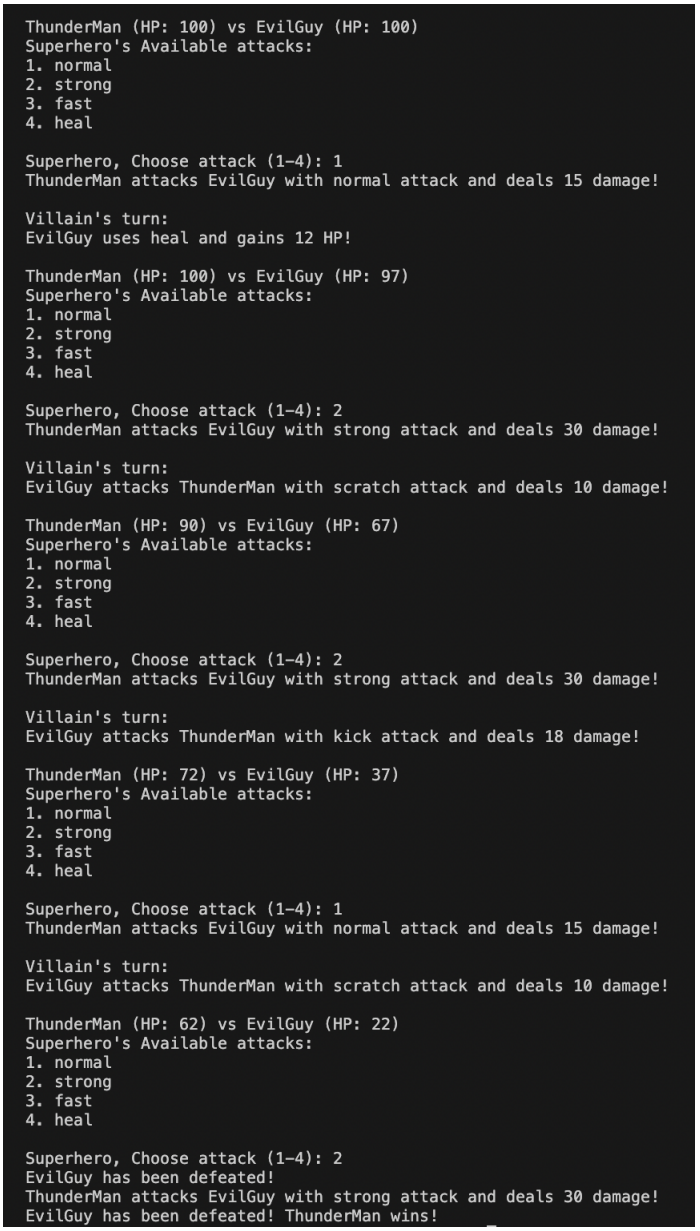
Tugas Anda pada lab kali ini adalah sebagai berikut:

- Buatlah dua kelas, yaitu Villain (penjahat) dan Superhero (pahlawan), masing-masing memiliki atribut seperti nama, health points (HP), dan berbagai jenis serangan dengan damage yang berbeda dalam menyerang lawan.
- Nama dari karakter superhero dan villain bisa dikustomisasi
- Setiap karakter Villain atau Superhero yang diciptakan awalnya memiliki health points penuh, yakni 100.
- Karakter Superhero dan Villain dapat memilih opsi heal yang akan meningkatkan health points mereka sehingga tidak akan menyerang lawan pada giliran tersebut.
- Hanya action (attack/heal) yang dilakukan oleh Superhero yang dapat

dikontrol melalui input

- Health points lawan akan berkurang sesuai dengan jenis serangan yang diberikan oleh Villain atau Superhero.
- Pertarungan akan terus berlanjut selama health points dari Villain dan Superhero masih lebih dari 0.

Contoh interaksi:



```
ThunderMan (HP: 100) vs EvilGuy (HP: 100)
Superhero's Available attacks:
1. normal
2. strong
3. fast
4. heal

Superhero, Choose attack (1-4): 1
ThunderMan attacks EvilGuy with normal attack and deals 15 damage!

Villain's turn:
EvilGuy uses heal and gains 12 HP!

ThunderMan (HP: 100) vs EvilGuy (HP: 97)
Superhero's Available attacks:
1. normal
2. strong
3. fast
4. heal

Superhero, Choose attack (1-4): 2
ThunderMan attacks EvilGuy with strong attack and deals 30 damage!

Villain's turn:
EvilGuy attacks ThunderMan with scratch attack and deals 10 damage!

ThunderMan (HP: 90) vs EvilGuy (HP: 67)
Superhero's Available attacks:
1. normal
2. strong
3. fast
4. heal

Superhero, Choose attack (1-4): 2
ThunderMan attacks EvilGuy with strong attack and deals 30 damage!

Villain's turn:
EvilGuy attacks ThunderMan with kick attack and deals 18 damage!

ThunderMan (HP: 72) vs EvilGuy (HP: 37)
Superhero's Available attacks:
1. normal
2. strong
3. fast
4. heal

Superhero, Choose attack (1-4): 1
ThunderMan attacks EvilGuy with normal attack and deals 15 damage!

Villain's turn:
EvilGuy attacks ThunderMan with scratch attack and deals 10 damage!

ThunderMan (HP: 62) vs EvilGuy (HP: 22)
Superhero's Available attacks:
1. normal
2. strong
3. fast
4. heal

Superhero, Choose attack (1-4): 2
EvilGuy has been defeated!
ThunderMan attacks EvilGuy with strong attack and deals 30 damage!
EvilGuy has been defeated! ThunderMan wins!
```

Anda **wajib** melengkapi template berikut dalam pengerjaan lab.

```
import random

class Superhero:
    #lengkapi kode ini
        self.name = name
        self.hp = 100
```

```
        self.attack_types = ["normal", "strong", "fast", "heal"]
        self.attack_damages = [15, 30, 12, 0]  # Fixed damage for Superhero's
attacks
        self.heal_amount = 15  # Fixed healing amount for Superhero

    def attack(self, villain, attack_index):
        if attack_index < 0 or attack_index >= len(self.attack_types):
            print("Invalid attack index!")

        damage = self.attack_damages[attack_index]
        villain.take_damage(damage)
        print(f"{self.name} attacks {villain.name} with
{self.attack_types[attack_index]} attack and deals {damage} damage!")


        if self.attack_types[attack_index] == "heal":
            self.heal()
            return

    def heal(self):
        self.hp = min([100, self.hp + self.heal_amount])
        print(f"{self.name} uses heal and gains {self.heal_amount} HP!")

    def take_damage(self, damage):
        self.hp -= damage
        if self.hp <= 0:
            print(f"{self.name} has been defeated!")

class Villain:
    def __init__(self, name):
        self.name = name
        self.hp = 100
        self.attack_types = ["scratch", "kick", "punch", "heal"]
        self.attack_damages = [10, 18, 15, 0]  # Fixed damage for Villain's
attacks
        self.heal_amount = 12  # Fixed healing amount for Villain

    def action(self, superhero):
        attack_index = random.randint(0, len(self.attack_types) - 1)
        random_action = #lengkapi kode ini

        if random_action != "heal":
            damage = #lengkapi kode ini
            superhero.take_damage(damage)
            print(f"{self.name} attacks {superhero.name} with
{self.attack_types[attack_index]} attack and deals {damage} damage!")

        else:
            #lengkapi kode ini

    def heal(self):
        self.hp = min([100, self.hp + self.heal_amount])
        print(f"{self.name} uses heal and gains {self.heal_amount} HP!")
```

```
    def take_damage(self, damage):
        self.hp -= damage
        if self.hp <= 0:
            print(f"{self.name} has been defeated!")

# Creating instances of Superhero and Villain
thunderman = Superhero("ThunderMan")
villain = Villain("EvilGuy")

# Game loop
while thunderman.hp > 0 and villain.hp > 0:
    print(f" \n{thunderman.name} (HP: {thunderman.hp}) vs {villain.name} (HP:
{villain.hp})")

    # Superhero's turn
    print("Superhero's Available attacks:")
    for i, attack_type in enumerate(thunderman.attack_types):
        print(f"{i + 1}. {attack_type}")
    superhero_attack_index = int(input("\nSuperhero, Choose attack (1-4): ")) - 1
    #lengkapi kode ini

    # Check if villain is defeated
    if villain.hp <= 0:
        print(f"{villain.name} has been defeated! {thunderman.name} wins!")
        break

    # Villain's turn (choosing a random action: attack or heal)
    print("\nVillain's turn:")
    villain.action(thunderman)

    # Check if Superhero is defeated
    if thunderman.hp <= 0:
        print(f"{thunderman.name} has been defeated! {villain.name} wins!")
        break
```

Deliverables

Anda **dilarang** keras melakukan plagiarisme! Tidak boleh memakai ChatGPT dan sejenisnya. Kumpulkan berkas .py yang berisi kode solusi Anda dengan format penamaan: `MOOCUI2023_DDP1_[NamaLengkap]_Lab03.py`

Contoh (jangan lupa ganti nama lengkapnya untuk submisi Anda):
`MOOCUI2023_DDP1_FarizDarari_Lab03.py`

Acknowledgement

- FD
- AD
- flaticon

DDP 1 - Lab 4

BookMan

Selamat datang di Dasar-Dasar Pemrograman 1 (DDP 1)!

Selamat datang di PyPrints Book Co, toko buku di Kota Codeburg yang menyediakan ragam ilmu pengetahuan seputar pemrograman dan teknologi. Namun, di balik kesenangan membaca, terdapat kekhawatiran akan kesalahan dalam manajemen stok dan harga. Kota ini membutuhkan seorang ahli untuk memastikan semua berjalan dengan lancar.

Muncullah seorang pengelola yang handal yaitu BookMan, bertugas untuk memelihara sistem informasi di toko tersebut. Tugasnya tak ringan, dia harus memastikan bahwa setiap buku tercatat dengan benar, stoknya selalu terjaga, dan harga setiap buku sesuai. Namun, tantangan terbesarnya adalah membangun *unit test* yang andal untuk memvalidasi setiap fungsi yang ada. *Unit test* merupakan kerangka kerja pengujian dalam Python yang digunakan untuk menulis dan menjalankan skenario pengujian pada perangkat lunak guna memastikan bahwa kode berfungsi sebagaimana yang diharapkan. Anda dapar menonton penjelasan terkait unit testing lebih lanjut disini: https://www.youtube.com/watch?v=HQ2Hg6HuqP0.

Tugasmu adalah membantu BookMan untuk menyiapkan unit test yang akan memeriksa apakah fungsionalitas dari sistem toko buku bekerja dengan baik. Mari kita bersama-sama memastikan PyPrints Book Co tetap menjadi tempat yang nyaman dan teratur bagi pencinta buku seputar teknologi dan pemrograman!

Tugas Anda pada lab kali ini adalah sebagai berikut:

- Buatlah sebuah class unit test bernama “TestBookstore”
- Memastikan bahwa fungsi calculate_total_price menghitung harga total dari

suatu jenis buku sesuai dengan kuantitasnya.

- Memastikan bahwa fungsi sell_book memberikan output "Insufficient stock." apabila jumlah suatu buku yang ingin dibeli melebihi stok yang tersedia.
- Memastikan bahwa fungsi restock_book memberikan output yang benar melakukan update stock buku yang tersedia di inventory.
- Memastikan bahwa fungsi remove_book juga mengatasi kesalahan (*error handling*) apabila ingin menghapus suatu data buku yang memang tidak tersedia pada inventory dengan mengeluarkan output "Book not found in inventory."
- Sebagai awalan, Anda dapat memanfaatkan 2 objek buku yang sudah ditambah pada fungsi setUp yang merupakan bawaan dari unittest untuk digunakan pada fungsi unit test lainnya.

Pastikan program dijalankan dan hasilnya adalah lima tes yang berhasil dieksekusi dalam waktu singkat, terlihat seperti berikut:

Anda **wajib** melengkapi template berikut dalam pengerjaan lab.

```
import unittest

class Bookstore:
    def __init__(self):
        self.inventory = {}

    def add_book(self, title, quantity, price):
        self.inventory[title] = {'quantity': quantity, 'price': price}

    def check_stock(self, title):
        return self.inventory.get(title, {'quantity': 0})['quantity']

    def calculate_total_price(self, title, quantity):
        book_info = self.inventory.get(title)
        if book_info:
            return book_info['price'] * quantity
        return 0

    def sell_book(self, title, quantity):
        if title in self.inventory and self.inventory[title]['quantity'] >=
quantity:
            self.inventory[title]['quantity'] -= quantity
            return f"{quantity} {title}(s) sold."
        return "Insufficient stock."
```

```
    def restock_book(self, title, quantity):
        if title in self.inventory:
            self.inventory[title]['quantity'] += quantity
            return f"{quantity} {title}(s) restocked."
        return "Book not found."

    def remove_book(self, title):
        if title in self.inventory:
            del self.inventory[title]
            return f"{title} removed from inventory."
        return "Book not found in inventory."

#Lengkapi kode

    def setUp(self):
        self.bookstore = Bookstore()  # Creating an instance of Bookstore
        self.bookstore.add_book("Python Crash Course", 10, 25)
        self.bookstore.add_book("Data Science Handbook", 5, 35)

    def test_check_stock(self):
        self.assertEqual(self.bookstore.check_stock("Python Crash Course"), 10)

    def test_calculate_total_price(self):
        #Lengkapi kode

    def test_sell_book(self):
        self.assertEqual(self.bookstore.sell_book("Python Crash Course", 3), "3
Python Crash Course(s) sold.")
        #Lengkapi kode

    def test_restock_book(self):
        #Lengkapi kode

    def test_remove_book(self):
        self.assertEqual(self.bookstore.remove_book("Data Science Handbook"),
"Data Science Handbook removed from inventory.")
        #Lengkapi kode

if __name__ == '__main__':
    unittest.main()
```

Deliverables

Anda **dilarang** keras melakukan plagiarisme! Tidak boleh memakai ChatGPT dan sejenisnya. Kumpulkan berkas .py yang berisi kode solusi Anda dengan format penamaan: `MOOCUI2023_DDP1_[NamaLengkap]_Lab04.py`

Contoh (jangan lupa ganti nama lengkapnya untuk submisi Anda):
`MOOCUI2023_DDP1_FarizDarari_Lab04.py`

Acknowledgement

- FD
- AD
- flaticon