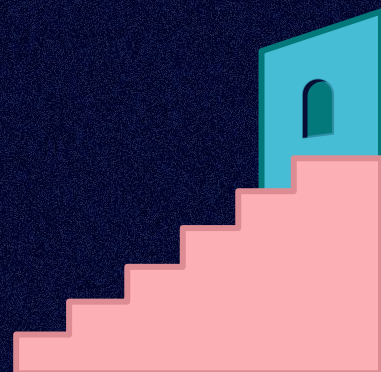
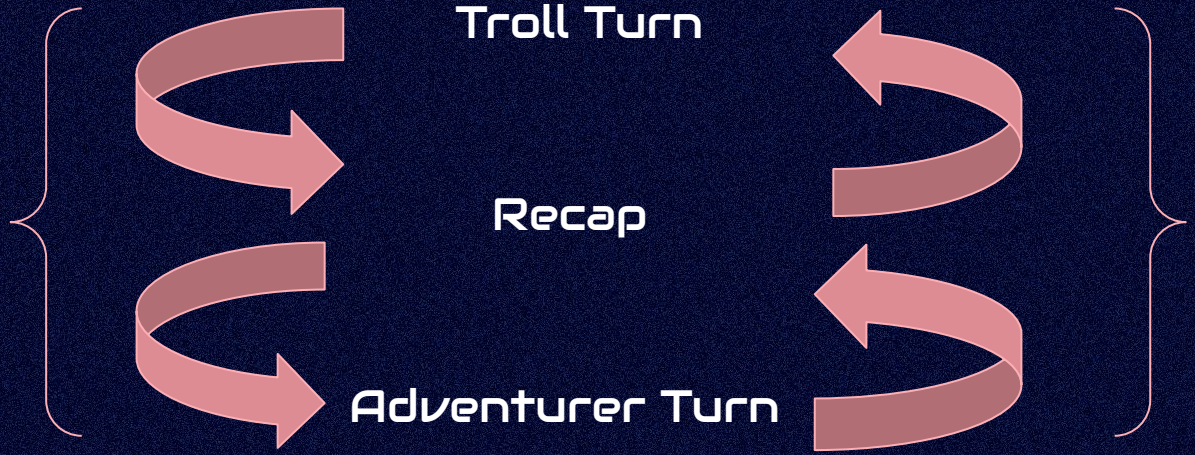
ArdaErincOguz_T1A3
Terminal Application

Fight The Troll



Feature 1: Turn-Based-Battle

While both characters have > 0 HP



Feature 1: Turn-Based-Battle

```python
while adventurer.health > 0 and troll.health > 0:
    try:
        Troll.troll_turn(troll, adventurer)
        if adventurer.health > 0 :
            recap(troll, adventurer)
        else:
            print('You have been defeated... Care to try again?')
            break
        Adventurer.adventurer_turn(adventurer, troll)
        if troll.health > 0 :
            recap(troll, adventurer)
        else:
            print('You have defeated the Troll, Congratulations!')
            break
```

The source code file battle.py contains the main loop, simply calling each turn until one instance reaches 0 health.

Output in Terminal

Troll Turn

Adventurer Turn

```
Arda, Your encounter begins, fighting a Troll in the wild!
The Cave Troll Swings his leg at you!
Critical Hit!
The Cave Troll deals 65 damage
The Cave Troll HP: 400
Arda HP: 85
What do you want to do? (Input selected option number)
1. Attack
2. Drink Potion of Healing
3. Cast Potency of Mana: Lasts 3 Turns
4. Escape
1
How would you like to attack?(Input selected option number)
1. Cast Fireball: 65 - 80 Base Fire Damage
2. Cast Lightning Tendrils: 60 - 90 Base Lightning Damage
3. Cast Life Drain: 60 - 80 Base Dark Magic Damage, Heals Self For 12 - 16 HP
1
Arda Casts fireball, englufing their enemy in flames
You deal 79 damage
The Cave Troll HP: 321
Arda HP: 85
The Cave Troll Swings his leg at you!
The Cave Troll deals 29 damage
The Cave Troll HP: 321
Arda HP: 56
What do you want to do? (Input selected option number)
1. Attack
2. Drink Potion of Healing
3. Cast Potency of Mana: Lasts 3 Turns
4. Escape
```

Health Recaps

Feature 2: Object Stats & their role

Adventurer or Troll Class(Hit Points (HP), Damage Multiplier, Critical Chance, Buff Counter)

Hit Points are the health of each class, and dictate whether the game keeps running or ends.

The Damage Multiplier stat will increases the damage done by attacks

The Critical Chance stat dictates whether an attack will critically hit, dealing significantly more damage

Integral to keep buff abilities temporary, this parameter takes effect and counts down each turn after a buff is activated

Feature 2: Object Stats & their role

```python
def __init__(self, name, health, dmg_mult, crit_chance, buff_counter):
    self.name = name
    self.health = health
    self.dmg_mult = dmg_mult
    self.crit_chance = crit_chance
    self.buff_counter = buff_counter
```

```python
        #Troll - Buff Counter
        if self.buff_counter > 0:
            self.dmg_mult = 1.3
            self.crit_chance = 0.35
            self.buff_counter -= 1

        else:
            self.dmg_mult = 1
            self.crit_chance = 0.2
```

```python
#Troll Move Randomizer
    def random_action(self):
        move_set = ['Throw Stone', 'Kick', 'Ground-Slam']
        move = random.choice(move_set)

    #Attack Move-Set
        if move == 'Throw Stone':
            troll_damage = random.randint(20,30) * self.dmg_mult
            print(f'{self.name} Hurls a nearby stone at you!')
        elif move == 'Ground-Slam':
            troll_damage = random.randint(15,30) * self.dmg_mult
            print(f'{self.name} Slams the ground around them, Upheavi
        elif move == 'Kick':
            troll_damage = random.randint(25,35) * self.dmg_mult
            print(f'{self.name} Swings his leg at you!')

    #Critical Chance Functionality
        if random.random() < self.crit_chance:
                sleep(1)
                troll_damage *= 1.8
                print('Critical Hit!')
```

Each Stat impacts the battle either by directly influencing the damage done, or influencing other stats.

Feature 3: Move Sets

Each Troll Turn:

- Attack:
  - Stone Throw
  - Ground Slam
  - Kick
- *AND*
  - *Chance to Roar (Buff)*
  - *Chance to eat nearby Morsel (Heal)*

Automated

```python
#Troll Move Randomizer
    def random_action(self):
        move_set = ['Throw Stone', 'Kick', 'Ground-Slam']
        move = random.choice(move_set)

    #Attack Move-Set
        if move == 'Throw Stone':
            troll_damage = random.randint(20,30) * self.dmg_mult
            print(f'{self.name} Hurls a nearby stone at you!')
        elif move == 'Ground-Slam':
            troll_damage = random.randint(15,30) * self.dmg_mult
            print(f'{self.name} Slams the ground around them, Upheaving the earth beneath you')
        elif move == 'Kick':
            troll_damage = random.randint(25,35) * self.dmg_mult
            print(f'{self.name} Swings his leg at you!')

    #Critical Chance Functionality
        if random.random() < self.crit_chance:
                sleep(1)
                troll_damage *= 1.8
                print('Critical Hit!')

    #Damage Output and Return
        sleep(1)
        print(f'{self.name} deals {int(troll_damage)} damage')
        return int(troll_damage)

#Troll -  Roar & Self Heal
    def buff_heal(self):
        if random.random() < 0.15 and self.buff_counter == 0:
            print("The Troll Roars, preparing to crush it's foe (Buff Lasts 3 Turns)")
            self.buff_counter = 3
            sleep(1.5)
        elif random.random() < 0.1 and self.buff_counter > 0:
            self.health += 50
            sleep(1.5)
            print('The troll devours a nearby morsel, restoring some of its health')
```

Feature 3: Move Sets

Each Adventurer Turn:

- Attack:
  - Fireball
  - Lightning Tendrils
  - Life Drain
- Heal
- Cast Buff (Lasts 3 Turns)
- Retreat (Exit Game)

Choice of one

Feature 3: Adventurer Move Sets

```python
def adventurer_turn(self, other):

    # Buff Counter
    if self.buff_counter > 0:
        self.dmg_mult = 1.3
        self.crit_chance = 0.35
        self.buff_counter -= 1

    else:
        self.dmg_mult = 1
        self.crit_chance = 0.2
    #Move Select
    try:
        selected_move = self.move_select()
        if selected_move == '1':
            #Attack
            adventurer_damage = self.attack()
            other.health -= adventurer_damage
            adventurer_healing = self.leech()
            self.health += adventurer_healing
        elif selected_move == '2':
            #Self-Heal
            adventurer_healing = self.selfheal()
            self.health += adventurer_healing
            #Buff
        elif selected_move == '3':
            print('Your mana surges, temporarily amplifying th
            self.buff()
            sleep(1)
            #exit
        elif selected_move == '4':
            print('You narrowly escape into the wilderness')
            exit()
```

```python
# Move Select
    def move_select(self):
        print('What do you want to do? (Input selected option number)')
        print('1. Attack')
        print('2. Drink Potion of Healing')
        print('3. Cast Potency of Mana: Lasts 3 Turns')
        print('4. Escape')

    #Move Selector
        try:
            selected_move = input()

        #Attack - Access to Move-Set
            if selected_move == ('1' or '2' or '3' or '4'):
                return selected_move

        #Error for invalid option
            else: raise Invalid_Input_Error
```

```python
    def attack(self):
        print('How would you like to attack?(Input selected option number)')
        sleep(1.5)
        print('1. Cast Fireball: 65 - 80 Base Fire Damage')
        print('2. Cast Lightning Tendrils: 60 - 90 Base Lightning Damage')
        print('3. Cast Life Drain: 60 - 80 Base Dark Magic Damage, Heals Self For 12 - 16 HP')
        global move
        global damage
        try:
            move = input()
    #Attack Moveset
            if move == '1' :
                print(f'{self.name} Casts fireball, englufing their enemy in flames')
                damage = random.randint(65,80) * self.dmg_mult

            elif move == '2' :
                print('Lighting reaches down from the heavens, broadly striking the area around y
                damage = random.randint(60,90) * self.dmg_mult

            elif move == '3' :
                print(f'Channeling more desperate, darker methods, {self.name} shaves off the lif
                damage = random.randint(60,80) * self.dmg_mult
            #Error for Invalid Option
            else: raise Invalid_Input_Error

        except KeyboardInterrupt:···

        except Invalid_Input_Error:···


    #Critical Chance Functionality
        if random.random() < self.crit_chance:···
        sleep(1.5)

    #Damage Output and Return
        print(f'You deal {int(damage)} damage')
        sleep(1)
        return int(damage)

    #Life-steal function for Life-Drain move
    def leech(self):···


#Healing Potion
    def selfheal (self):
```

All Together Now

Adventurer Options

Adventurer Attack - Life Drain

Health Recap

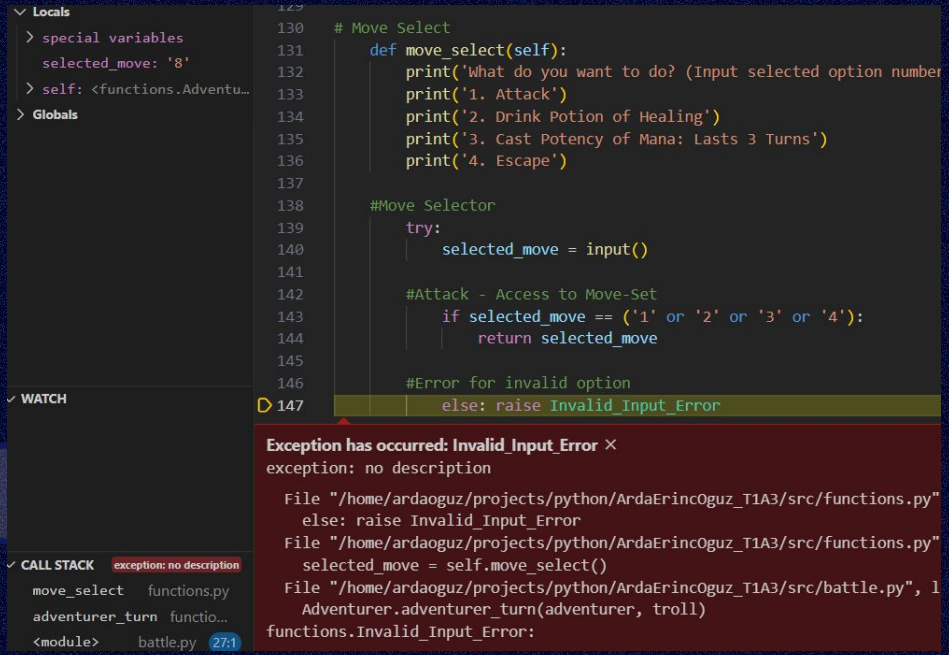
Adventurer Buff

Troll Critical Hit

Adventurer Heal

```
What do you want to do? (Input selected option number)
1. Attack
2. Drink Potion of Healing
3. Cast Potency of Mana: Lasts 3 Turns
4. Escape
1
How would you like to attack?(Input selected option number)
1. Cast Fireball: 65 - 80 Base Fire Damage
2. Cast Lightning Tendrils: 60 - 90 Base Lightning Damage
3. Cast Life Drain: 60 - 80 Base Dark Magic Damage, Heals Self For 12 - 16 HP
3
Channeling more desperate, darker methods, Arda shaves off the life of their enemy, adding to their own
You deal 67 damage
You heal yourself for 13 health
The Cave Troll HP: 333
Arda HP: 122
The Cave Troll Hurls a nearby stone at you!
The Cave Troll deals 30 damage
The Cave Troll HP: 333
Arda HP: 92
What do you want to do? (Input selected option number)
1. Attack
2. Drink Potion of Healing
3. Cast Potency of Mana: Lasts 3 Turns
4. Escape
3
Your mana surges, temporarily amplifying the effects of your atacks
The Cave Troll HP: 333
Arda HP: 92
The Cave Troll Slams the ground around them, Upheaving the earth beneath you
Critical Hit!
The Cave Troll deals 48 damage
The Cave Troll HP: 333
Arda HP: 44
What do you want to do? (Input selected option number)
1. Attack
2. Drink Potion of Healing
3. Cast Potency of Mana: Lasts 3 Turns
4. Escape
2
You heal yourself for 44 health
```

Challenges & Issues



- Testing functions requiring inputs that then called other functions
  - Had to reformat the entire code and pull apart different parts into separate functions to be able to test them
- Some tests for raising errors didnt seem to work despite errors raised

```
monkeypatch = <_pytest.monkeypatch.MonkeyPatch object at 0x7f010ac5d690>

    def test_raise_error_move_select(monkeypatch):
        move_input = (StringIO('8')) #mocking invalid input
        monkeypatch.setattr('sys.stdin', move_input)
>       with pytest.raises(Exception):
E       Failed: DID NOT RAISE <class 'Exception'>

src/input_testing.py:12: Failed
----------------------------------------------------------------------- Captured stdout call -----
What do you want to do? (Input selected option number)
1. Attack
2. Drink Potion of Healing
3. Cast Potency of Mana: Lasts 3 Turns
4. Escape
Invalid Option
=========================================================== short test summary info =====
FAILED src/input_testing.py::test_raise_error_move_select - Failed: DID NOT RAISE <class 'Exception'>
============================================================== 1 failed in 1.02s ========
(.venv) ardaoguz@AEO-Work-Laptop:~/projects/python/ArdaErincOguz_T1A3$
```

Favourite Parts

```
def choose_move(self):
    print('What would you like to do?')
    print('1. Cast Fireball')
    print('2. Cast another Fireball')
    print('3. Cast Yeet-Fireball')
    move = input()
    if move == '1' :
        print('I cast fireball')
        damage = random.randint(10,20) * self.dmg_mult
        print(f'You inflict {damage} damage on your enemy')
    elif move == '2' :
        damage = random.randint(15,30) * self.dmg_mult
        print(' Oh look, another spell slot. I cast fireball')
        print(f'Your enemy is engulfed in flame, dealing {damage} damage on your enemy')
    elif move == '3' :
        damage = random.randint(40,60) * self.dmg_mult
        print(' I didnt ask how big the room is, I said, I yeet a big-ass fireball')
        print(f'you deal {damage} damage to your enemy. There is no sign remaining of your enemy')
```

Thank You!