```python
"""
To make the core battle logic more concise, we can use lambda functions.
"""


class Character:
    """
    A class representing a character in a game.
    can be Barbarian, Wizard, Paladin, etc.
    """

    def __init__(self, hp, attack, defend):
        self.hp = hp
        self.attack = attack
        self.defend = defend


# no class DamageCalculator here anymore


class BattleManagerWithLambda:
    # using lambda function we can make the code more concise
    # and the main logic is steady and will not be modified frequently
    def deal_damage(attacker, defender, damage_calculator, args):
        if damage_calculator:
            defender.hp -= damage_calculator(attacker, defender, args)
```

```python
# example of using the code
def main():
    attacker = Character(hp=100, attack=50, defend=20)
    defender = Character(hp=100, attack=30, defend=10)

    # deal damage using minus method
    BattleManagerWithLambda.deal_damage(
        attacker, defender,
        lambda attacker, defender, args: attacker.attack - defender.defend,
        []
    )

    # deal damage using times method
    BattleManagerWithLambda.deal_damage(
        attacker, defender,
        lambda attacker, defender, args: round(
            attacker.attack * (
                defender.defend * 1.0 / (args[0] + defender.defend)
            )
        ),
        [10]
    )

    # deal damage using true damage method
    BattleManagerWithLambda.deal_damage(
        attacker, defender,
        lambda attacker, defender, args: args[0],
        [10]
    )
```