הנדסת תוכנה
שנקר
מהנד0ים טאלנטים בת1כנה

תרגיל 3

Operator Overloading && Inheritance

חובה:

- הגשה ליחידים בלבד
- התוכנית חייבת להתקמפל
- לרשום את השמות שלכם ואת תעודות הזהות בראש כל קובץ בעזרת comments

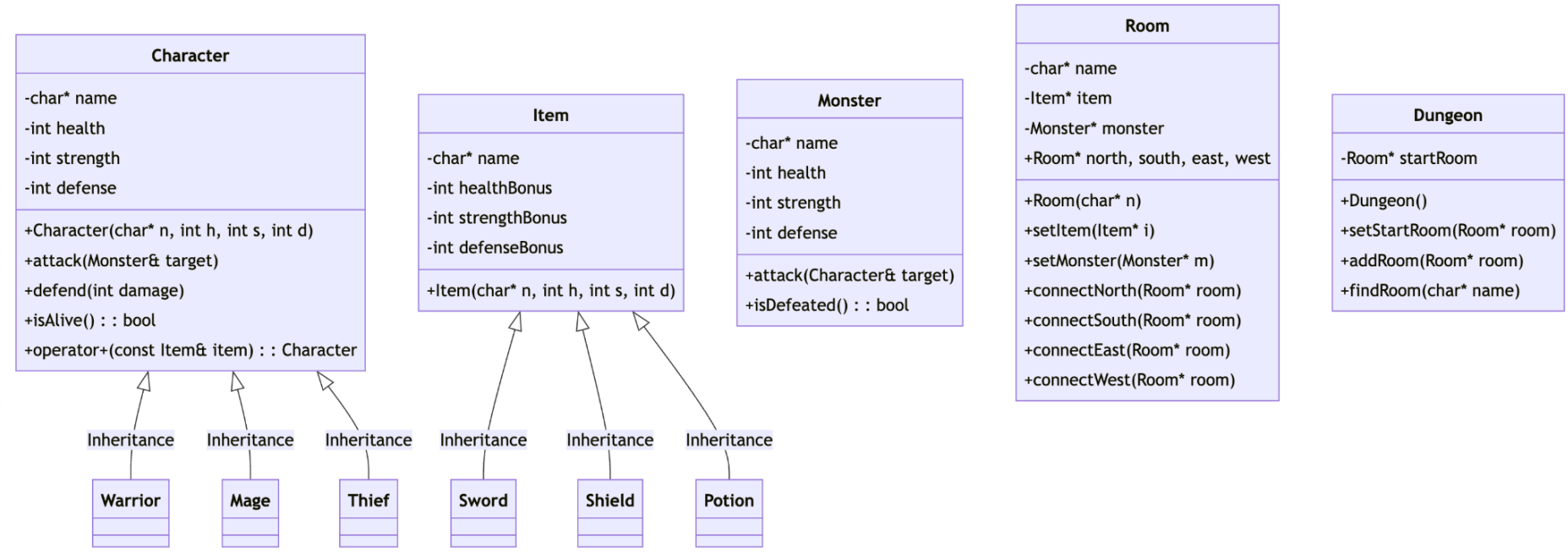
דגשים:

- לא לשכוח לבצע בדיקות ל null כשצריך
- לא לשכוח לשחרר זיכרון לאחר שסיימתם עם הקצאת זיכרון או קובץ פתוח
- לתת שמות משמעותיים למשתנים
- הערות (Comments) מעל כל פונקציה המתארות מה היא עושה

מטרת התרגיל:

בתרגיל זה תתרגלו את שני הנושאים העיקריים שלמדתם, **Operator Overloading and Inheritance**
יהיה עליכם לבנות מחלקות ואובייקטים על פי ההנחיות מטה, אנא וודאו כי הם עומדים בדרישות!

עליכם לכתוב את המבנה הבא בקוד:



חשוב: הוספת Constructors, Destructors, Getters and Setters היא לגמרי בידיים שלכם ובהתאם לצורך.

הסבר על המחלקות (החלק הזה יהיה באנגלית כי זה מקפיץ את השמות בעברית):

Character Class (Base Class)

- Members:
    - char* name: Character's name.
    - int health: Character's health points.
    - int strength: Character's strength, affecting combat outcomes.
    - int defense: Character's defense, mitigating incoming damage.
- Functions:
    - Character(char* n, int h, int s, int d): Constructor to initialize character attributes.
    - void attack(Monster& target): Simulates attacking a monster.
    - void defend(int damage): Reduces health based on incoming damage, considering defense.
    - bool isAlive() const: Checks if the character's health is above 0.
    - Character operator+(const Item& item): Overloads the + operator to apply item effects to the character.

Item Class

- Members:
    - char* name: Item's name.
    - int healthBonus: Bonus health provided by the item.
    - int strengthBonus: Bonus strength provided by the item.
    - int defenseBonus: Bonus defense provided by the item.
- Functions:
    - Item(char* n, int h, int s, int d): Constructor to initialize item attributes.

Monster Class

- Members:
    - char* name: Monster's name.
    - int health: Monster's health points.
    - int strength: Monster's strength.
    - int defense: Monster's defense.
- Functions:
    - void attack(Character& target): Simulates attacking a character.
    - bool isDefeated() const: Checks if the monster's health is 0 or less.

Room Class Explanation

- Members:
    - char* name: Identifier for the room.
    - Room* north, south, east, west: Connections to adjacent rooms.
    - Item* item: Item in the room, if any.
    - Monster* monster: Monster in the room, if any.
- Functions:
    - Room(char* n): Constructor to initialize the room with its name and set pointers (north, south, east, west, item, monster) to nullptr.
    - void connectNorth(Room* room): Sets the northern connection.
    - void connectSouth(Room* room): Sets the southern connection.
    - void connectEast(Room* room): Sets the eastern connection.
    - void connectWest(Room* room): Sets the western connection.
    - void setItem(Item* i): Places an item in the room.
    - void setMonster(Monster* m): Places a monster in the room.

Dungeon Class Explanation

- Members:
    - Room* startRoom: The starting point of the dungeon.
    - Room** rooms: Pointer to an array of room pointers, managing all rooms in the dungeon.
- Functions:
    - Dungeon(): Constructor to initialize the dungeon, setting startRoom to nullptr and preparing the rooms array.
    - void setStartRoom(Room* room): Defines the starting room of the dungeon.
    - void addRoom(Room* room): Adds a room to the dungeon's rooms array.
    - Room* findRoom(char* name): Searches for and returns a room by its name.

Explanation of Input Commands

- **Create Character/Room:** Initializes characters and rooms with specified attributes.
- **Set StartRoom:** Designates the starting room for the dungeon exploration.
- **Add Room:** Adds a room to the dungeon's layout.
- **Connect:** Establishes directional connections between rooms.
- **Place Item/Monster:** Places items and monsters in specified rooms.
- **Enter Dungeon:** Starts the character's journey in the dungeon.
- **Move:** Moves the character to an adjacent room in the specified direction.
- **Fight:** Engages the character in combat with a room's monster.
- **PickUp:** Adds an item from the room to the character's inventory.

Explanation of Numbers in the Input File

Numbers in the text.txt input file represent the attributes for characters, items, and monsters. For example:

- “Create Character Warrior Thorin 100 15 10” creates a character named Thorin with 100 health, 15 strength, and 10 defense.
- “Place Item Sword 0 5 0” places a Sword item that provides a 0 health bonus, 5 strength bonus, and 0 defense bonus.

Calculation Example

When a character picks up an item, their stats are updated based on the item's bonuses. For instance, if Thorin picks up a Sword with a +5 strength bonus, his strength increases from 15 to 20.

Damage Calculation: Monster's Strength (20) - Character's Defense (15) = 5.

Key Movement Keywords

- **Room Connections:** Connect <Room1> <Room2> <Direction> establishes a navigable link between two rooms in the specified direction. Directions include North, South, East, and West.
- **Character Movement:** Move <Character> <Direction> commands the character to move in the specified direction from their current location. The game engine checks if a connection exists in that direction and updates the character's location accordingly.

הנדסת תוכנה
שנקר
מהנד0ים טאלנטים בת1כנה

Game

סיפור רקע:

אתם יוצאים להרפתקה!
עליכם להיכנס למקום המסוכן ביותר בממלכת שנ-חם (יעני שנ-קר), ולנצח את המפלצות הארורות ששומרות על האוצרות הנדירים ביותר בתבל!
חיזרו עם שלל וקבלו את אהדת העם כולו!

The D&D logo

קווים מנחים:

- הבדיקה תעשה על ידי הרצת Main והכנסת קובץ output על ידי הבודק
- עליכם לחשוב מחוץ לקופסא להתמודד עם מקרי קצה שיכולים לקרות שקוראים לפונקציות שלכם
- כל פעם שמפלצת מנצחת את הגיבור, יורד לו חיים לפי ההפרש של ההתקפה שלה לעומת ההגנה שלו

דרישות:

- כל מחלקה בקובץ CPP ו h משלה
- המשחק רץ בעזרת ה Main ואין מגע אדם בזמן ריצה

Good luck,
Yuval Ozeri

yuvalozeri@shenkar.ac.il