Middle East Technical University | Department of Computer Engineering

CENG 242

PROGRAMMING LANGUAGE CONCEPTS

Spring '2022-2023

PE5: The Band Wars

June 5, Monday, 2023

1 Introduction

And then it was time... For the Band Wars 2023!

We got rock bands with catchy riffs. We got metal bands with eye-popping solos. We got jazz quartets with groove. And we don't really know much about K-Pop but we got them, too.

But this is no music festival with opening acts and headliners. No, Band Wars pits two bands against each other! They play to the same crowd, and one shall emerge victorious! The fans, understandably, leave their old allegiance and join the victors after each battle.

Each band has to join to the Tournament where only one can emerge victorious. The rules are simple, once they lose, they lose. They will walk away with shame and the only one of them will be the Ultimate Champion.

Your job is to help each band type to use their special abilities and decide the winner, the ultimate, the unstoppable champion.

2 Getting Started

This homework is designed to measure your knowledge about the most common object-oriented concepts such as Inheritance, polymorphism, and abstraction. Your job is to create a tournament where 4 different types of bands compete. The band types are:

- RockBand
- JazzBand
- MetalBand
- KPopBand

3 Bands

All of these bands are a **MusicBand** which is your base class for all of the bands.

3.1 MusicBand

A MusicBand has following attributes:

- name (const char [])(private) : Name of the band. Uniquely identifies a band.
- fans (int)(private) : Number of fans of a band. Changes during future events are going to be explained.
- energy (int)(protected) : Energy of the band. Changes during future events are going to be explained.
- talent (int)(protected) : Skill level of a band. Differs from band to band. Changes during future events are going to be explained.

Moreover, a MusicBand has the following methods:

- virtual int get_fan_count() final : Getter of the fans attribute. Will be used by the grader.
- virtual void set_fan_count(int fans) final : Setter of the fans attribute. Will be used by the grader.
- virtual int get_talent() final : Getter of the talent attribute. Will be used by the grader.
- virtual void set_talent(int talent) final : Setter of the talent attribute. Will be used by the grader.
- virtual const string& get_name() final : Getter of the name attribute. Will be used by the grader.
- virtual const void rename(const string& name) final : Setter of the fans attribute. Will be used by the grader.
- virtual int get_energy() final: Getter of the energy attribute. Will be used by the grader.
- virtual void set_energy(int n) final: Setter of the energy attribute. Will be used by the grader.
- virtual double play(MusicBand& other) = 0: A competition between two teams. Each band will have a different play method.
- virtual double rehearse() = 0 : Virtual function of rehearse method that should be implemented in all derived functions.

All of the functions of MusicBand are implemented **beforehand**. You should be given a copy of this class and you are **not allowed** to change the given implementation (at each run your version will be changed with the default one).

The methods of the MusicBand generally consist of getters and setters, which are implemented for you. However, you should override **play** and **rehearse** methods in the child classes and each band has a different version of the play and the rehearse methods.

3.1.1 Play Method

virtual int play(MusicBand *other)

The play method is returning an int score value and has only one parameter which is another MusicBand. You can think of this method as a band playing against another. In the play method, only the LHS of the method has a chance

to play and gets a score. The score calculation is simple and given below:

$$Score = (fans + 0.1 * talent * energy) * C, \tag{1}$$

where C is a constant change by the type of the bands.

The fans, talent, and energy are the values of the band given in the MusicBand class. The constant C is defined for pairs of bands which is the main difference between a KPopBand and a MetalBand. There is a table given below that explains the C constant. The left side of the table shows the LHS of the play method while the top of table 1 shows the other band in the method.

	KPopBand	MetalBand	RockBand	JazzBand
KPopBand	2.0	0.5	0.5	0.5
MetalBand	0.5	1.0	1.5	1.1
RockBand	0.5	1.4	1.0	0.8
JazzBand	0.5	1.3	0.7	0.7

Table 1: Constant definition for each band pair

Moreover, each band type uses a different energy amount during play action. The used energy is calculated by a constant percentage. The calculation is given below:

$$remaining\_energy = energy - energy * K \tag{2}$$

where K is different for each band and given in table 2.

KPopBand	0.2
MetalBand	0.16
RockBand	0.1
JazzBand	0.06

Table 2: Energy usage percentages

Using these values you should calculate (using Equation 1) the score of the band and return the value. Moreover, you should change the energy of the current band by using energy usage calculation as in Equation 2. During the execution of this method, **nothing happens except these two operations**.

3.1.2 Rehearse Method

virtual void rehearse(void)

Similar to the play method, rehearsing is different for each band type. After the rehearse method is called, each band loses some amount of energy and gains talent. The energy change is given by the formula 3

$$energy = energy - (0.5 * K) \tag{3}$$

where K values are given in table 2 (same with the play method).

Moreover, their talent change formula is given in Equation 4.

$$talent = talent + T \tag{4}$$

where T is given in table 3.

	Talent Change
KPopBand	0
MetalBand	-5
RockBand	10
JazzBand	5

Table 3: Talent Change After Rehearse

Important!
All of the implementations of 4 bands should be implemented by you. Each band should inherit the MusicBand base class.

4 Tournament

Now, after the implementation of the Bands, the challenge is to prepare a Tournament to see which Band is the best. For that purpose, you will be given two classes which are:

- Tournament
- TournamentRound

4.1 Tournament Class

The Tournament class is already implemented for you and you should finish the implementation of TournamentRound class to prepare a proper Tournament. The Tournament class gets a set of MusicBands and using the **single elimination Tournament system**, decides the winner of the Tournament. The single elimination method takes a group of teams and by matching them 1v1 eliminates one of them until only 1 team remains. The implementation of the Tournament class is given below.

Each Tournament has a name and a list of MusicBands. **Similar to the MusicBand class this class will be reloaded by the original implementation before the execution of your solution**. Moreover, there are two methods that help running the Tournament:

- void enroll(MusicBand& band): Adds a new team to the bands in the Tournament.
- void make_tournament(): The main functionality of the Tournament class. Runs the tournament by calling TournamentRound class until only one team is left in the bands of the Tournament.

Tournament

```
void Tournament::enroll(MusicBand&team)
{
    bands.push_back(&team);
}

void Tournament::make_tournament()
{
    TournamentRound current_bands(bands);
    TournamentRound rounds;
    int round_number = 1;

    std::cout << "printing round " << round_number++ << std::endl
                                   << current_bands << std::endl;
    while (current_bands.size() > 1) {
        current_bands = std::move(current_bands.get_next_round());
        std::cout << "printing round " << round_number++ << std::endl
                                       << current_bands << std::endl;
    }
}
```

4.2 TournamentRound

The TournamentRound class runs each round of the single elimination method. It has only one attribute which is bands competing in that round. These bands are initially bands in the Tournament class and then after the first round halves the size (because of the single elimination method). Moreover, there are 7 more methods of the class which are given below:

- TournamentRound(): Default constructor of the TournamentRound.
- TournamentRound(std::vector<MusicBand*> bands): The constructor of the TournamentRound which takes a group of bands.
- std::size_t size(): Returns the size of the bands.
- TournamentRound(TournamentRound& other): Copy constructor of the TournamentRound.
- TournamentRound(TournamentRound&& other): Move constructor of the TournamentRound.
- TournamentRound& operator=(TournamentRound&& other): Overloads the = operator.
- TournamentRound& get_next_round(): The main functionality of the TournamentRound. Explained in detail in the section 4.2.1
- friend std::ostream& operator<< (std::ostream &os, TournamentRound &other): Overloads the << operator. Used by the Tournament class to print the current band names in the TournamentRound, separated by a single tab.

4.2.1 Get Next Round

The method that is used to run each round. For the bands in that round, matches two teams compete against each other, one from the front and one from the back, starting from the first and the last band. Moreover, returns the winners as a new TournamentRound object. For example, assume there are 4 bands on a particular TournamentRound:

- BLACKPINK
- Nevermore
- ShePassedAway
- JohnColtrane

There will be two competitions on that round between BLACKPINK-JohnColtrane and Nevermore-ShePassedAway. Moreover, the method returns a new TournamentRound containing the winners. Assume JohnColtrane wins against BLACKPINK and Nevermore wins against ShePassedAway, then you should return a TournamentRound with bands JohnColtrane and Nevermore. The order in the list is important. Since you will pick 2 teams one from the front and one from the back, the BLACKPINK-JohnColtrane match occurs before the other. So, JohnColtrane should be added to the resulting TournamentRound before Nevermore.

A match is completed by calling the play method of the competitors one-by-one. The play method returns a score value as explained in Section 3.1.1. The band with a bigger score value wins that TournamentRound. Moreover, bands steal fans from each other. The winning band steals a certain amount of fans from the losing band. The fan change is calculated using the given Equation 5

$$fan\_change = |Score_1 - Score_2| \tag{5}$$

Note that, you can not steal fans more than a band has. In that case, you should steal all the remaining fans from that band. The example below explains the 1v1 competition:

Assume BLACKPINK has 100 fans, 100 talent and 80 energy and it is a KPopBand, and Nevermore has 80 fans, 140 talent and 90 energy and it is a MetalBand. Using the play method of the KPopBand it gets a score:

$Score_1$ = (100 + 0.1 * 100 * 80) * 0.5 = 450

Then, MetalBand plays against the KPopBand and gets the score:

$Score_2$ = (80 + 0.1 * 140 * 90) * 0.5 = 670

The MetalBand wins the competition (added to the winners of the TournamentRound) and steals 670-450 = 220 fans from the KPopBand. However, since the KPopBand only has 100 fans, MetalBand gets all of them and now has 180 fans while the KPopBand has 0 fans.

Note: If there is an odd number of bands inside a TournamentRound, the middle team (the last remaining one) will directly move towards to the next TournamentRound.

5 Regulations

1. **Implementation and Submission:** The template header files are available in the Virtual Programming Lab (VPL) activity called "PE5" on OdtuClass. You can either download the files and work on your local setup or use the VPL itself to implement your solution.
2. **Cheating:** We have zero tolerance policy for cheating. People involved in cheating (e.g. code similarity with others or AI models) will be punished according to the university regulations.

3. **Grading:** Your codes will be auto-graded by using the black-box grading technique. Moreover, you will be given a set of sample test cases and there will be test cases on the VPL that is grading your solution. After the deadline, a new set of test cases will be applied to test your solutions.
4. **Grade Elements:** You will be graded by full implementation of 4 MusicBand classes and TournamentRound class. There will be cases that only evaluate MusicBands in case you are not able to implement TournamentRound class. However, you should complete the implementation of the MusicBand classes in order to get a grade from the TournamentRound implementation.

6 Tips

1. You should follow OOP principles since the homework is designed to measure your knowledge of OOP. If you follow these steps, you will have an easy time completing the homework.
2. Similar to the PE4, you will need to use dynamic casting to check the type of a particular MusicBand. However, you can follow other options for that purpose, it's up to you.

7 Set Lists

Each band has a set of sample songs given by the competitor bands to help to on your journey. While these songs do not affect the grading, you might listen to them while doing the homework.

7.1 RockBand

A set of sample songs that are played by the RockBands are given below:

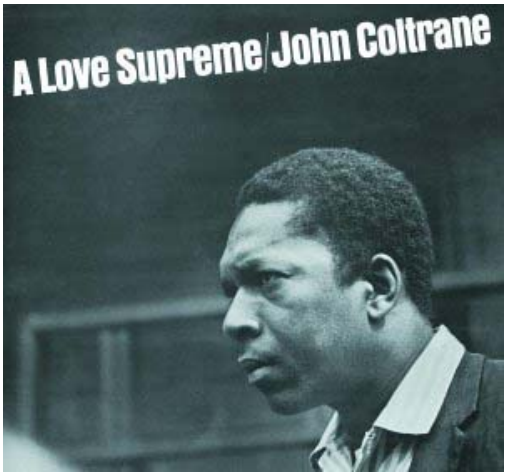
- Nirvana - Lithium
- Perl Jam - Black
- Foo Fighters - The Pretender
- Red Hot Chili Peppers - Californication
- Arctic Monkeys - Do I Wanna Know?
- Muse - Supremacy
- King Gizzard & the Lizard Wizard - Ice V
- Barış Manço - Dönence

7.2 MetalBand

A set of sample songs that are played by the MetalBands are given below:

- Iron Maiden - Paschendale
- Megadeth - Symphony Of Destruction
- Iron Maiden - Aces High
- Judas Priest - Angel
- Motörhead - Ace of Spades
- Slayer - War Ensemble
- Black Sabbath - Iron Man
- Rammstein - Mein Herz Brennt
- Nevermore - Dreaming Neon Black
- Nevermore - The Heart Collector
- Warrel Dane - Brother

7.3 JazzBand



A set of sample songs that are played by the JazzBands are given below:

- John Coltrane - A Love Supreme
- Edith Piaf - La vie en rose
- Duke Ellington - Caravan
- Mouse on the keys - Seiren
- Emily Remler - Tenor Madness
- Hiromi Uehara - The Tom and Jerry Show
- Miles Davis - So What
- Chet Baker - Almost blue
- Ibrahim Maalouf - True Story
- Keith Jarrett - THE KÖLN CONCERT

7.4 KPopBand

A set of sample songs that are played by the KPopBands are given below:

- BTS - Dynamite
- Blackpink - How You Like That
- Bigbang - Bang Bang Bang
- Red Velvet - Russian Roulette
- Exo - Love Shot
- TXT - Sugar Rush Ride
- Seventeen - Hot
- GOT7 - Just Right