In task 3, I measured the runtime of copying versus moving a Player in a Guild to another Guild when one Player has a 10x10-size Inventory of Items versus another Player with a 2000x2000-size Inventory of items. After running my testing program, the output is:

Copying 10x10 Inventory Player Runtime: 4 microseconds
Moving 10x10 Inventory Player Runtime: 0 microseconds

Copying 2000x2000 Inventory Runtime: 95872 microseconds
Moving 2000x2000 Inventory Runtime: 1 microseconds

The results show that the difference between copying and moving one player with a 10x10 Inventory is minimal (4 ms vs less than 0 ms) because the small Player object does not have a large enough Inventory of Items to make a significant difference. However, there is a significant difference in time efficiency between copying and moving a player with a 2000x2000 Inventory (95872 ms vs 1 ms). This is because moving objects through move semantics (l-values & r-values) doesn't create a full duplicate of the objects. This saves meaningful time, as evident by the output. In this case, moving Players will be more efficient compared to copying Players since a player cannot be in two guilds at the same time. Copying a Player object between two Guilds would be unnecessary when you can transfer/move the Player from the old guild to the new guild.

My test program:

```
#include <iostream>
#include <chrono>
#include "Guild.hpp"
#include "Inventory.hpp"

int main() {
    // Intialize a player, smallPlayer with a small Inventory, smallInv
    Item sword("Sword", 5.0, WEAPON);

    Inventory smallInv;
    for (int r = 0; r < 10; ++r) {
        for (int c = 0; c < 10; ++c) {
            smallInv.store(r, c, sword);
        }
    }
    Player smallPlayer("SmallHero", smallInv);
```

```
    // Intialize a player, largePlayer with a large Inventory, largeInv
    Inventory largeInv(std::vector<std::vector<Item>>(2000,
std::vector<Item>(2000)));
    for (int r1 = 0; r1 < 2000; ++r1) {
        for (int c1 = 0; c1 < 2000; ++c1) {
            largeInv.store(r1, c1, sword);
        }
    }
    Player largePlayer("BigHero", largeInv);


    // Initialize a small guild and large guild for testing.
    Guild smallGuild;
    smallGuild.enlistPlayer(smallPlayer);
    Guild smallGuildtomove;
    Guild smallGuildtocopy;

    Guild largeGuild;
    largeGuild.enlistPlayer(largePlayer);
    Guild largeGuildtomove;
    Guild largeGuildtocopy;

    /*Copying Small Guild Test*/
    const auto copySmalltimeStart =
std::chrono::high_resolution_clock::now();
    smallGuild.copyPlayerTo("SmallHero", smallGuildtocopy);
    const auto copySmalltimeEnd =
std::chrono::high_resolution_clock::now();
    const auto copySmallusec =
std::chrono::duration_cast<std::chrono::microseconds>(copySmalltimeEnd -
copySmalltimeStart);
    std::cout << "Copying 10x10 Inventory Player Runtime: " <<
copySmallusec.count() << " microseconds" << std::endl;

    /*Moving Small Guild Test*/
    const auto moveSmalltimeStart =
std::chrono::high_resolution_clock::now();
    smallGuild.movePlayerTo("SmallHero", smallGuildtomove);
    const auto moveSmalltimeEnd =
std::chrono::high_resolution_clock::now();
```

```
    const auto moveSmallusec =
std::chrono::duration_cast<std::chrono::microseconds>(moveSmalltimeEnd -
moveSmalltimeStart);
    std::cout << "Moving 10x10 Inventory Player Runtime: " <<
moveSmallusec.count() << " microseconds" << std::endl;

    std::cout <<
"-------------------------------------------------------------------------
-------------------------------------------" << std::endl;

    /*Copying Small Guild Test*/
    const auto copyLargetimeStart =
std::chrono::high_resolution_clock::now();
    largeGuild.copyPlayerTo("BigHero", largeGuildtocopy);
    const auto copyLargetimeEnd =
std::chrono::high_resolution_clock::now();
    const auto copyLargeusec =
std::chrono::duration_cast<std::chrono::microseconds>(copyLargetimeEnd -
copyLargetimeStart);
    std::cout << "Copying 2000x2000 Inventory Runtime: " <<
copyLargeusec.count() << " microseconds" << std::endl;

    /*Moving Small Guild Test*/
    const auto moveLargetimeStart =
std::chrono::high_resolution_clock::now();
    largeGuild.movePlayerTo("BigHero", largeGuildtomove);
    const auto moveLargetimeEnd =
std::chrono::high_resolution_clock::now();
    const auto moveLargeusec =
std::chrono::duration_cast<std::chrono::microseconds>(moveLargetimeEnd -
moveLargetimeStart);
    std::cout << "Moving 2000x2000 Inventory Runtime: " <<
moveLargeusec.count() << " microseconds" << std::endl;

    return 0;
}
```