Q1-Q4 - Fix or improve the implementation of the below methods.

Q1 - Fix or improve the implementation of the below methods

```
local function releaseStorage(player)
player:setStorageValue(1000, -1)
end

function onLogout(player)
if player:getStorageValue(1000) == 1 then
addEvent(releaseStorage, 1000, player)
end
return true
end
```

S1 - Solution

```
local function releaseStorage(player)
player:setStorageValue(1000, -1)
end

function onLogout(player)
if player:getStorageValue(1000) ~= -1 then  -- Release when the value is not equal to -1
addEvent(releaseStorage, 1000, player)
end
return true
end
```

Q2 - Fix or improve the implementation of the below method

```
function printSmallGuildNames(memberCount)
-- this method is supposed to print names of all guilds that have less than memberCount max members
local selectGuildQuery = "SELECT name FROM guilds WHERE max_members < %d;"
local resultId = db.storeQuery(string.format(selectGuildQuery, memberCount))
local guildName = result.getString("name")
print(guildName)
end
```

S2 - Solution

```
function printSmallGuildNames(memberCount)

-- declare the Query and store it
local selectGuildQuery = "SELECT name FROM guilds WHERE max_members < %d;"
local resultId = db.storeQuery(string.format(selectGuildQuery, memberCount))

-- iterate through the resulting query and print each name
if resultId ~= false then
repeat
```

```
local guildName = result.getString(resultId,”name”)
print(guildName)
until not result.next(resultId)
result.free(resultId)
end
end
```

Q3 - Fix or improve the name and the implementation of the below method

```
function do_sth_with_PlayerParty(playerId, membername)
player = Player(playerId)
local party = player:getParty()

for k,v in pairs(party:getMembers()) do
if v == Player(membername) then
party:removeMember(Player(membername))
end
end
end
```

S3 - Solution

```
function doRemoveMemberPlayerParty(guid, membername)

local player = Player(guid)
local member = Player(membername)

-- check if Player is partying
if not isInParty(guid) then
return false
end

-- select the player’s party and get its Members
local party = player:getParty()
local partyMembers = party:getMembers()

-- search for the member in the party, and remove it if it is founded
for _,v in pairs(partyMembers) do
if v:getGuid() == member:getGuid() then
party:removeMember(member)
end
end
end
```

Q4 - Assume all method calls work fine. Fix the memory leak issue in below method

```
void Game::addItemToPlayer(const std::string& recipient, uint16_t itemId)
{
Player* player = g_game.getPlayerByName(recipient);
if (!player) {
```

```
player = new Player(nullptr);
if (!IOLoginData::loadPlayerByName(player, recipient)) {
return;
}
}

Item* item = Item::CreateItem(itemId);
if (!item) {
return;
}

g_game.internalAddItem(player->getInbox(), item, INDEX_WHEREEVER, FLAG_NOLIMIT);

if (player->isOffline()) {
IOLoginData::savePlayer(player);
}
}
```

S4 - Solution

```
void Game::addItemToPlayer(const std::string& recipient, uint16_t itemId)
{
Player* player = g_game.getPlayerByName(recipient);   //  Assign pointer to player

//If there is no player we create one
if (!player) {
player = new Player(nullptr);

// Logad the Player Data, in case we can't do it, we will exit and free the memory
if (!IOLoginData::loadPlayerByName(player, recipient)) {
delete player;
return;
}
}

//Create the item, in case we can't do it, we will exit and free the memory
Item* item = Item::CreateItem(itemId);
if (!item) {
delete player;
return;
}
g_game.internalAddItem(player->getInbox(), item, INDEX_WHEREEVER, FLAG_NOLIMIT);

if (player->isOffline()) {
IOLoginData::savePlayer(player);
}
}
```