

継承を使った演習問題

- C++作業フォルダ内に**SampleRPG**フォルダを作成
`mkdir SampleRPG`
`cd SampleRPG`
- **SampleRPG**フォルダ内に `chara.h`, `chara.cpp`,
`player.h`, `player.cpp`, `main.cpp` の7つの
ファイルを作成する
`copy nul chara.h`
`copy nul chara.cpp`
(略)

継承を使った演習問題

- `chara.h`に`Chara`クラスを定義する
- `protected`なメンバ変数(すべて`int`型)
`m_Hp`, `m_Atk`, `m_Def` を定義する
- 各メンバ変数に対して、`public`な`ゲッター`を定義する
(※セッターは不要)。
- コンストラクタは3つの引数(`m_Hp`, `m_Atk`, `m_Def`)
をもち、引数の値をメンバ変数の初期値にする
- デフォルトコンストラクタではメンバ変数を0にする
- 各関数の内容は`chara.cpp`に記述する

継承を使った演習問題

- `player.h`に`Chara`クラスを継承した`Player`クラスを定義する
- `private`なメンバ変数 `m_Sp`(`int`型) を追加で定義
- `public`なゲッターも併せて定義する(※セッター不要)
- コンストラクタは、4つの引数(`m_Hp`, `m_Atk`, `m_Def`, `m_Sp`)をもち、引数の値をメンバ変数の初期値にする
- デフォルトコンストラクタはメンバ変数を0にする
- 各関数の内容は`player.cpp`に記述する。

継承を使った演習問題

Charaクラス

```
public:  
    getHp()  
    getAtk()  
    getDef()  
protected:  
    m_Hp  
    m_Atk  
    m_Def
```

Playerクラス

```
public:  
    getSp()  
private:  
    m_Sp
```

- ・親クラスと子クラスで
 コンストラクタは2種類(引数あり、なし)
- ・デストラクタは使用しない

継承を使った演習問題

• main.cpp (SampleRPG)

```
#include "chara.h"
#include "player.h"
#include <iostream>
using namespace std;
int main() {                                //Hp   Atk Def   Sp
    Player* pPlayer = new Player(100, 50, 20, 30);
    cout << "Playerの状態" << endl
         << "  HP  :" << pPlayer->getHp() << endl
         << "  SP  :" << pPlayer->getSp() << endl
         << " Atk:" << pPlayer->getAtk() << endl
         << " Def:" << pPlayer->getDef() << endl;
    delete pPlayer;
    return 0;
}
```

Charaクラス

```
public:
    getHp()
    getAtk()
    getDef()
protected:
    m_Hp
    m_Atk
    m_Def
```

Playerクラス

```
public:
    getSp()
private:
    m_Sp
```

vectorを使った演習問題

- `main.cpp`を変更する
- `vector<Player*> pPlayer` を宣言
Playerクラスのコンテナインスタンス
`pPlayer`を生成して、初期値として
`new Player(100, 50, 20, 30)`
を与えるように変更する
- さらに`push_back()`にて
`new Player(300, 70, 40, 50)`
のインスタンスを追加したあと、すべてのコンテナの要素
のメンバを表示する

実行結果

Playerの状態

HP: 100

SP: 30

Atk: 50

Def: 20

Playerの状態

HP: 300

SP: 50

Atk: 70

Def: 40

継承を使った演習問題

• main.cpp (SampleRPG)


```
#include "chara.h"
#include "player.h"
#include <iostream>
→ #include <vector> //ヘッダファイルの追加
using namespace std;
int main() {
    //Hp  Atk  Def  Sp
    Player* pPlayer = new Player(100, 50, 20, 30);
    cout << "Playerの状態" << endl
         << "  HP  :" << pPlayer->getHp() << endl
         << "  SP  :" << pPlayer->getSp() << endl
         << "  Atk:" << pPlayer->getAtk() << endl
         << "  Def:" << pPlayer->getDef() << endl;
    delete pPlayer;
```

継承を使った演習問題

• main.cpp (SampleRPG)

```
#include "chara.h"
#include "player.h"
#include <iostream>
#include <vector>
```

```
using namespace std;
int main() {  
    //vectorの宣言とPlayerインスタンスの追加
```



```
    vector<Player*> pPlayer{ new Player(100, 50, 20, 30) };  
    cout << "Playerの状態" << endl  
        << "  HP  :" << pPlayer->getHp() << endl  
        << "  SP  :" << pPlayer->getSp() << endl  
        << "  Atk:" << pPlayer->getAtk() << endl  
        << "  Def:" << pPlayer->getDef() << endl;  
    delete pPlayer;
```


継承を使った演習問題

• main.cpp (SampleRPG)

```
#include "chara.h"
#include "player.h"
#include <iostream>
#include <vector>
using namespace std;
int main() {
    vector<Player*> pPlayer{ new Player(100, 50, 20, 30) };
    pPlayer.push_back(new Player(300, 70, 40, 50));
    // データ末尾にPlayerインスタンスを追加
    << " SP : " << pPlayer->getSp() << endl
    << " Atk:" << pPlayer->getAtk() << endl
    << " Def:" << pPlayer->getDef() << endl;
    delete pPlayer;
}
```

継承を使った演習問題

• main.cpp (SampleRPG)

```
int main() {  
    vector<Player*> pPlayer{ new Player(100, 50, 20, 30) };  
    pPlayer.push_back(new Player(300, 70, 40, 50));  
    for(int i = 0; i < pPlayer.size() ; i++){  
        // 配列要素の個数ぶんだけループ  
        cout << "HP : " << pPlayer[i]->getHp() << endl  
              << " SP : " << pPlayer[i]->getSp() << endl  
              << " Atk:" << pPlayer[i]->getAtk() << endl  
              << " Def:" << pPlayer[i]->getDef() << endl;  
    }  
    delete pPlayer;  
    return 0;  
}
```

継承を使った演習問題

• main.cpp (SampleRPG)

```
int main() {  
    vector<Player*> pPlayer{ new Player(100, 50, 20, 30) };  
    pPlayer.push_back(new Player(300, 70, 40, 50));  
    for(int i = 0; i < pPlayer.size() ; i++){  
        cout << "Playerの状態" << endl  
            << "  HP  :" << pPlayer[i]->getHp() << endl  
            << "  SP  :" << pPlayer[i]->getSp() << endl  
            << "  Atk:" << pPlayer[i]->getAtk() << endl  
            << "  Def:" << pPlayer[i]->getDef() << endl;  
    }  
    delete pPlayer;  
    return 0;  
}
```

// 配列の添え字を追加

継承を使った演習問題

• main.cpp (SampleRPG)

```
int main() {  
    vector<Player*> pPlayer{ new Player(100, 50, 20, 30) };  
    pPlauer.push back(new Plauer(300, 70, 40, 50));  
    for  
        {  
            delete pPlayer;  
            return 0;  
        }
```

使用を終了したインスタンスは消去する必要がある

そのための手順は

①vectorの要素に格納したインスタンスを消去

②vectorの要素自体を削除

となるため、単にdeleteするだけではダメ



継承を使った演習問題

• main.cpp (SampleRPG)

```
for(int i = 0; i < pPlayer.size() ; i++){  
    cout << "Playerの状態" << endl  
        << "  HP  :" << pPlayer[i]->getHp() << endl  
        << "  SP  :" << pPlayer[i]->getSp() << endl  
        << "  Atk:" << pPlayer[i]->getAtk() << endl  
        << "  Def:" << pPlayer[i]->getDef() << endl;  
    ,
```

//先頭要素を指すイテレータを定義

```
auto itr = pPlayer.begin();  
delete pPlayer;  
return 0;
```

```
}
```

継承を使った演習問題

• main.cpp (SampleRPG)


```
for(int i = 0; i < pPlayer.size() ; i++){  
    cout << "Playerの状態" << endl  
        << "  HP  :" << pPlayer[i]->getHp() << endl  
        << "  SP  :" << pPlayer[i]->getSp() << endl  
        << "  Atk:" << pPlayer[i]->getAtk() << endl  
        << "  Def:" << pPlayer[i]->getDef() << endl;  
}  
auto itr = pPlayer.begin();  
while( itr != pPlayer.end() ){  
    delete pPlayer;  
}  
return 0;  
}
```

//最後尾までループ

継承を使った演習問題

• main.cpp (SampleRPG)

```
for(int i = 0; i < pPlayer.size() ; i++){
    cout << "Playerの状態" << endl
         << "  HP  :" << pPlayer[i]->getHp() << endl
         << "  SP  :" << pPlayer[i]->getSp() << endl
         << "  Atk:" << pPlayer[i]->getAtk() << endl
         << "  Def:" << pPlayer[i]->getDef() << endl;
}
auto itr = pPlayer.begin();
while( itr != pPlayer.end() ){
    delete *itr;
}
return 0;
}
```



//イテレータの示すアドレス
(Playerクラスのインスタンス)を解放

継承を使った演習問題

• main.cpp (SampleRPG)

```
for(int i = 0; i < pPlayer.size() ; i++){  
    cout << "Playerの状態" << endl  
        << "  HP  :" << pPlayer[i]->getHp() << endl  
        << "  SP  :" << pPlayer[i]->getSp() << endl  
        << "  Atk:" << pPlayer[i]->getAtk() << endl  
        << "  Def:" << pPlayer[i]->getDef() << endl;  
}  
auto itr = pPlayer.begin();  
while( itr != pPlayer.end() ){  
    delete *itr;  
    itr = pPlayer.erase(itr);  
}  
return 0;  
}
```

// 配列の要素も併せて削除する
要素の個数が変わるためイテレータを更新しておく