

new演算子とdelete演算子

- メモリ上の空き領域をプログラマの使いたいタイミングで確保したり、いらなくなった時点でメモリを解放することができる
- new演算子
必要なときに必要なぶんメモリを確保する
- delete演算子
不要になったメモリを解放する

メモリ領域

- プログラム領域

プログラム自体(機械語)を格納するメモリ領域

- 静的領域

グローバル変数を格納するメモリ領域

- ヒープ領域 ← new/deleteで使用可能

プログラム実行中、動的に確保されるメモリ領域

- スタック領域

ローカル変数を格納するメモリ領域

new演算子とdelete演算子

- 教科書P144~145 Sample402
- Sample401フォルダからSample402フォルダを作成
robocopy Sample401 Sample402
cd Sample402
- main.cppを編集

new演算子とdelete演算子

- main.cpp (Sample402)

```
int main() {  
    Car* pkuruma = nullptr;  
    pkuruma = new Car();  
    pkuruma->setSpeed(40);  
    pkuruma->drive(1.5);  
    pkuruma->setSpeed(60);  
    pkuruma->drive(2.0);  
    cout << "総走行距離:" << pkuruma->getMigration()  
          << "km" << endl;  
    delete pkuruma;  
    cout << "インスタンスの消去完了" << endl;  
    return 0;  
}
```

new演算子とdelete演算子

- コンパイルの仕方

コマンドプロンプトで次のコマンドを入力する

```
cl _/EHsc _main.cpp _car.cpp
```

成功したら、main.exeを実行して結果を確認

new演算子とdelete演算子

• main.cpp (Sample)

Carクラスのポインタ変数**pkuruma**を宣言して **nullptr** で初期化 (NULLでも可。中身がない(NULL:0)なポインタという意味で**nullptr**を使う)

```
int main() {  
    Car* pkuruma = nullptr;  
    pkuruma = new Car();  
    pkuruma->setSpeed(40);  
    pkuruma->drive(1.5);  
    pkuruma->setSpeed(60);  
    pkuruma->drive(2.0);  
    cout << "総走行距離:" << pkuruma->getMigration()  
          << "km" << endl;  
    delete pkuruma;  
    cout << "インスタンスの消去完了" << endl;  
    return 0;  
}
```

new演算子とdelete演算子

• main.cpp (Sample)

```
int main() {  
    Car* pkuruma = nullptr;  
    pkuruma = new Car();  
    pkuruma->setSpeed(40);  
    pkuruma->drive(1.5);  
    pkuruma->setSpeed(60);  
    pkuruma->drive(2.0);  
    cout << "総走行距離:" << pkuruma->getMigration()  
          << "km" << endl;  
    delete pkuruma;  
    cout << "インスタンスの消去完了" << endl;  
    return 0; }
```

new演算子を使って、コンストラクタ
を実行して、インスタンスを生成
ポインタ変数 **pkuruma** に
インスタンスのアドレスを代入

ここで指定するのはコンストラクタ名
でクラス名でないことに注意！

new演算子とdelete演算子

- main.cpp (Sample402)

```
int main() {  
    Car* pkuruma = nullptr;  
    pkuruma = new Car();  
    pkuruma->setSpeed(40);  
    pkuruma->drive(1.5);  
    pkuruma->setSpeed(60);  
    pkuruma->drive(2.0);  
    cout << "総走行距離:" << pkuruma->getMigration()  
          << "km" << endl;  
    delete pkuruma;  
    cout << "インスタンスの消去完了" << endl;  
    return 0; }
```

ポインタを用いる場合は「.」でなく
「->(アロー)」を使う

new演算子とdelete演算子

- main.cpp (Sample402)

```
int main() {  
    Car* pkuruma = nullptr;  
    pkuruma = new Car();  
    pkuruma->setSpeed(40);  
    pkuruma->drive(1.5);  
    pkuruma->setSpeed(60);  
    pkuruma->drive(2.0);  
    cout << "総走行距離  
         << "km" << endl;  
    delete pkuruma;  
    cout << "インスタンス  
    return 0; }
```

【重要!!】

new演算子を使って生成したインスタンスはdelete演算子を使って破棄しなければならない！(破棄しないとメモリリークが発生)

new演算子とdelete演算子

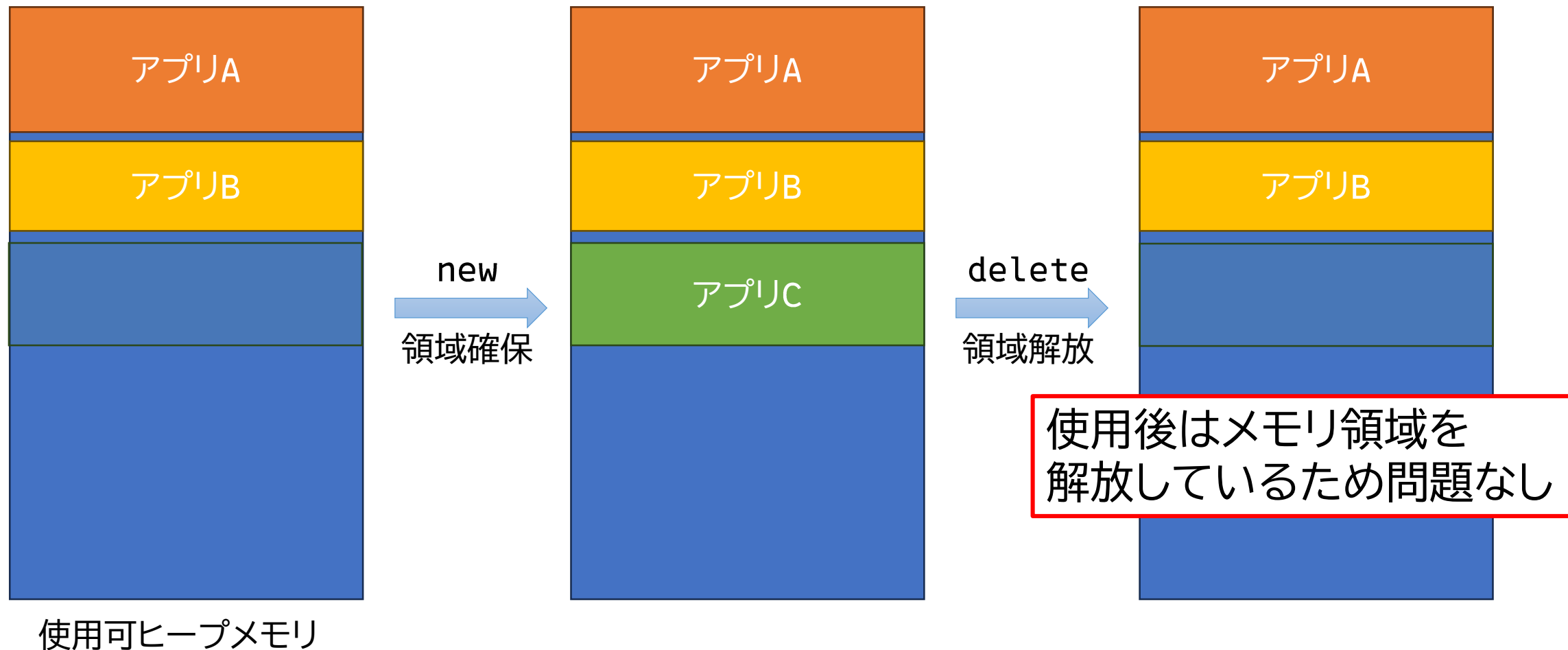
- メモリリーク

new演算子等で動的に確保したメモリ領域を使用後に解放せずに放置することで、使用できるメモリ領域が次第に減っていくこと

最終的には、システムが利用するためのメモリ領域もなくなるため、システムがハングアップしてしまうという危険性がある

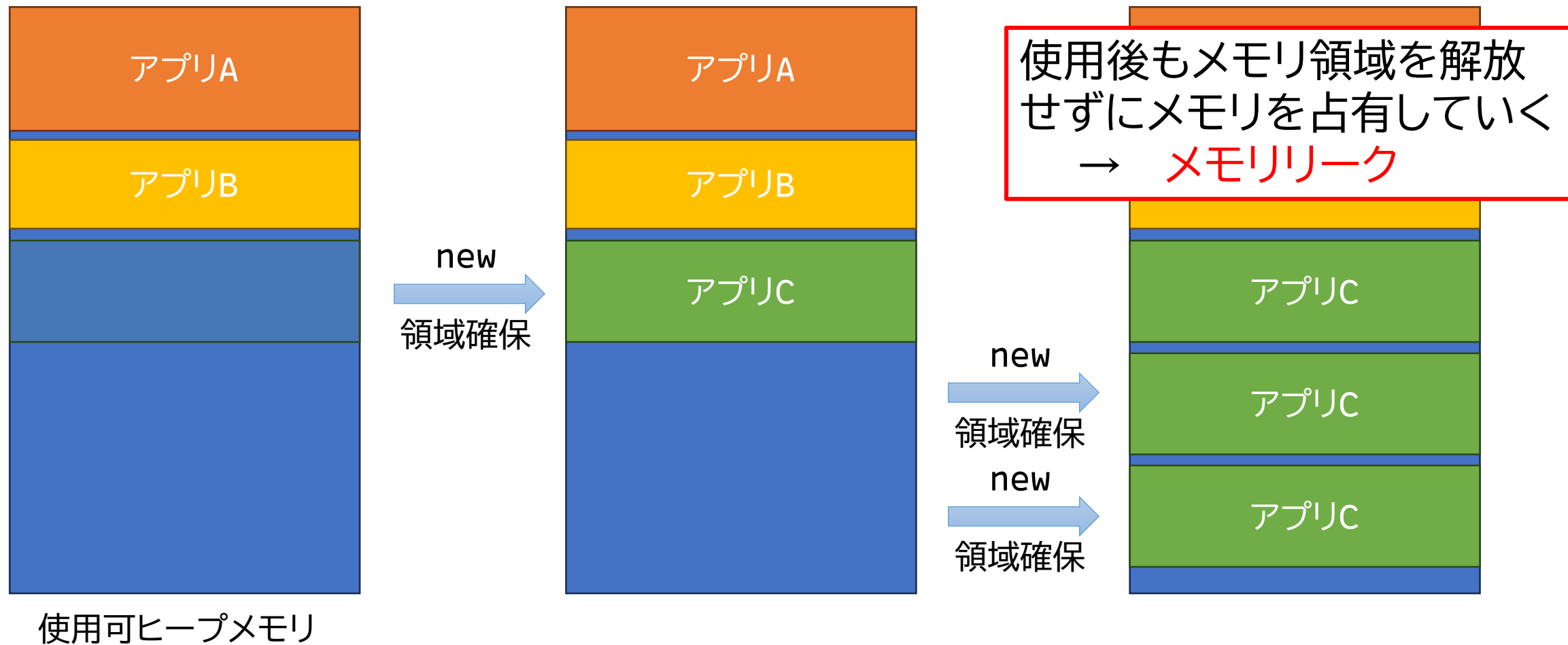
new演算子とdelete演算子

- メモリリーク



new演算子とdelete演算子

- メモリリーク



new演算子とdelete演算子

- 教科書P148 Sample403
- Sample403フォルダを作成
`mkdir Sample403`
`cd Sample403`
- main.cppを作成
`copy nul main.cpp`

new演算子とdelete演算子

- main.cpp (Sample403)

```
include<iostream>
using namespace std;
int main() {
    int* p = nullptr;
    p = new int();
    *p = 123;
    cout << *p << endl;
    delete p;
    return 0;
}
```

new演算子とdelete演算子

- main.cpp (Sample403)

```
include<iostream>
using namespace std;
int main() {
    int* p = nullptr;
    p = new int();
    *p = 123;
    cout << *p << endl;
    delete p;
    return 0;
}
```

int型のポインタ変数 **p** を宣言し
て**nullptr**で初期化
new int()で整数値を格納できる
メモリ領域を確保

使用していたメモリ領域
(アドレスは**p**に格納)を解放

new演算子とdelete演算子

- main.cpp (Sample403)

```
include<iostream>
using namespace std;
int main() {
    int* p = nullptr;
    p = new int(123); //領域を確保して数値で初期化
    //*p = 123;
    cout << *p << endl;
    delete p;
    return 0;
}
```


new演算子とdelete演算子

- 教科書P149 Sample404
- Sample404フォルダを作成
`mkdir Sample404`
`cd Sample404`
- main.cppを作成
`copy nul main.cpp`

new演算子とdelete演算子

- main.cpp (Sample404)

```
include<iostream>
using namespace std;
int main() {
    int* p = nullptr;
    p = new int[10];
    for(int i = 0; i < 10; i++){
        p[i] = i;
        cout << p[i] << " ";
    }
    cout << endl;
    delete[] p;
    return 0;
}
```

new演算子とdelete演算子

• main.cpp (Sample404)

```
include<iostream>
using namespace std;
int main() {
    int* p = nullptr;
    p = new int[10];
    for(int i = 0; i < 10; i++){
        p[i] = i;
        cout << p[i] << " ";
    }
    cout << endl;
    delete[] p;
    return 0;
}
```

int型のポインタ変数 **p** を宣言し
て**nullptr**で初期化
new int[10]で10個の整数値を格
納できる配列用メモリ領域を確保

配列の場合は**delete[]**として
メモリ領域を解放

new演算子とdelete演算子

- main.cpp (Sample404)

```
include<iostream>
using namespace std;
int main() {
    int* p = nullptr;
    p = new int[10];
    for(int i = 0; i < 10; i++){
        *(p + i) = i;
        cout << *(p + i) << " ";
    }
    cout << endl;
    delete[] p;
    return 0;
}
```

ポインタと配列要素との関係

new演算子とdelete演算子

- まとめ
- new演算子
必要なときに必要なぶんメモリを確保する
クラスのインスタンスだけでなく、基本型やその配列にも使用可能
- delete演算子
不要になったメモリを解放する
newしたものは必ずdeleteする必要がある！