C++ 007 コンストラクタ

【コンストラクタ】

コンストラクタはインスタンスの生成時に自動的に呼び出される特殊なメンバ関数です。

コンストラクタはメンバ関数名をクラス名と同じにします。性質上、引数は設定できますが戻り値は設定できません。void も記述しません。

```
class sample {
public:
    sample();           //コンストラクタの宣言
};

sample::sample() {      //コンストラクタの実装
    処理を記述
}

int main() {
    sample instance;    //インスタンス生成時に実行される
```

＜コンストラクタに引数を設定する＞

```
class personal {
public:

    //メンバ変数
    std::string name;    //名前
    int age;             //年齢

    //コンストラクタに引数を設定
    personal(std::string, int);
};

//コンストラクタの実装
personal::personal(std::string name, int age) {
    this->name = name;
    this->age = age;
}

int main() {
    personal instance("A 太郎", 20);    //コンストラクタに実引数
```

＜this＞

「this」は自インスタンスを示し「this->メンバ名」で使用します。

ローカル変数とメンバ変数の変数名が同じ場合、ローカル変数名が優先されてしまうため、メンバ変数には this を付けて区別します。

【デストラクタ】

デストラクタはインスタンスの破棄時に自動的に呼び出される特殊なメンバ関数です。

デストラクタはメンバ関数名を「~（チルダ）＋クラス名」にします。その性質上、引数も戻り値も設定できません。void も記述しません。

```
class sample {
public:
    ~sample();          //デストラクタの宣言
};

sample::~sample() {     //デストラクタの実装
    処理を記述
}
```

＜例文＞

```
class sample {
public:
    sample();     //コンストラクタの宣言
    ~sample();    //デストラクタの宣言
};

//コンストラクタの実装
sample::sample() {
    std::cout << "コンストラクタ発動" << std::endl;
}

//デストラクタの実装
sample::~sample() {
    std::cout << "デストラクタ発動" << std::endl;
}

int main() {
    sample instance;    //インスタンス生成時にコンストラクタが実行
    return 0;
}                       //auto 変数はここで破棄されるので、デストラクタが実行
```

＜実行結果＞

コンストラクタ発動

デストラクタ発動

【デフォルトコンストラクタ】

デフォルトコンストラクタは「自動的に定義される引数の無いコンストラクタ」です。

1 つもコンストラクタが定義されていないクラスには、デフォルトコンストラクタが定義されますが、1 つでもコンストラクタを定義したクラスには、デフォルトコンストラクタは定義されません。

下記の personal クラスには、自動的にデフォルトコンストラクタが定義されます。

```
class personal {
public:
    std::string name;
    int age;
};

int main() {
    personal ins;   //インスタンス化できる
```

下記の personal クラスには、デフォルトコンストラクタは定義されません。

よって引数無しではインスタンス化はできません。

```
class personal {
public:
    std::string name;
    int age;

    personal(std::string, int);    //引数付きコンストラクタ
};

personal::personal(std::string name, int age) {
    this->name = name;
    this->age = age;
}

int main() {
    personal instance1("A 太郎",20);    //インスタンス化できる
    personal instance2;                //インスタンス化できない。エラーになる。
```

引数無しでインスタンス化したい場合は、コンストラクタをオーバーロードして「引数無しのコンストラクタ」を定義する必要があります。

```
class personal {
public:
    std::string name;
    int age;

    personal();                     //引数無しのコンストラクタ
    personal(std::string, int);     //引数付きコンストラクタ
};

//引数無しのコンストラクタ
personal::personal() {
    //処理の記述は不要
}

//引数付きコンストラクタ
personal::personal(std::string name, int age) {
    this->name = name;
    this->age = age;
}

int main() {
    personal instance1("A 太郎",20);     //インスタンス化できる
    personal instance2;                 //インスタンス化できる
```

引数無しでインスタンス化するのに、引数無しのコンストラクタは定義するだけでいいです。必要無ければ処理は記述しなくてもかまいません。

【コピーコンストラクタ】

コピーコンストラクタはインスタンスをコピーするためのコンストラクタです。

コピーコンストラクタの記述がひとつも無いクラスには自動的に「デフォルトコピーコンストラクタ」が付きます。

```
class personal {
public:

    //メンバ変数
    std::string name;       //名前
    int age;                //年齢

    //メンバ関数の宣言
    void introduction();    //自己紹介する関数
};

//自己紹介する関数の実装
void personal::introduction() {
    std::cout << "私は" << name << "です。年齢は" << age << "歳です。" << std::endl;
}

//メイン関数
int main() {

    personal x;
    x.name = "A 太郎";
    x.age = 20;

    personal y(x);       //コピーコンストラクタが実行
    y.introduction();    //インスタンス x がコピーされている
```

＜実行結果＞

私は A 太郎です。年齢は 20 歳です。