Composition and initialization

- Version 1: Dr. Ofir Pele
- Version 2: Dr. Miri Ben-Nissan
- Version 3: Dr. Erel Segal-Halevi

Composition

In Java – pointers.

In C++ - objects.

Construction order:
  small to big.

Destruction order:
big to small.

The paramaterless ctor (aka **default ctor**)

```cpp
int main() {
    B b;
}
```

```cpp
class A {
public:
 A() {
   std::cout <<

   "A - " <<

   "parameterless" <<

   " ctor\n";
 }
};
```

```cpp
class B {
   A _a1,_a2;
public:
   B() {
     std::cout <<

     "B - parameterless " <<

     "ctor\n";
   }
};
```

```cpp
// output
A - parameterless ctor
A - parameterless ctor
B - parameterless ctor
```

The paramaterless ctor (aka **default ctor**)

```cpp
int main() {
    B b;
}
```

```cpp
class A {
public:
    A(int a) {
        std::cout <<
        "A ctor with one
        parameter\n";
    }
};
```

```cpp
class B {
    A _a1,_a2;
public:
    B() {
        std::cout <<
        "B – parameterless " <<
        "ctor\n";
    }
};
```

```cpp
// compilation error
No parameterless ctor for _a1, _a2
```

The initialization list

```cpp
int main() {
    B b(2,3);
}
```

```cpp
class A {
public:
    A(int a) {
        std::cout <<
        "A (" << a << ") "

        << std::endl;
    }
};
```

```cpp
class B {
    A _a1,_a2;
public:
    B(int i, int j)
    :_a1 (i), _a2(j)
    {
        std::cout
        << "B cons"
        << std::endl;
    }
};
```

```
// output
A (2)
A (3)
B cons
```

Initialization using pointers (1)

```cpp
int main() {
    B b(2);
}
```

```cpp
class A {
public:
    A(int a) {
        std::cout <<

        "A (" << a << ") "

        << std::endl;
    }
};
```

```cpp
class B {
    A *_ap;
public:
    B(int i);
};


B::B(int i) {
    _ap = new A (i);
    cout << "B cons\n";
}
```

```cpp
// output
A (2)
B cons
```

Initialization using pointers (2)

```cpp
int main() {
    B b(2);
}
```

```cpp
class A {
public:
    A(int a) {
        std::cout <<
        "A (" << a << ") "

        << std::endl;
    }
};
```

```cpp
class B {
    A *_ap;
public:
    B(int i);
};


B::B(int i)
    : _ap (new A(i))
{
    cout << "B cons\n";
}
```

```
output//
A (2)
B cons
```

The initialization list

```cpp
int main() {
    B b(2,3);
}
```

```cpp
class A {
public:
    A(int a) {
        std::cout <<
        "A (" << a << ") "

        << std::endl;
    }
};
```

```cpp
class B {
    A _a1,_a2;
public:
    B(int i, int j)
    :_a1 (i), _a2(j)
    {
        std::cout
        << "B cons"
        << std::endl;
    }
};
```

```
// output
A (2)
A (3)
B cons
```

The initialization list

1. Initialization of object members.
2. Initialization of constants and reference variables.
3. Initialization of parent class.
4. It is faster and safer to use the initialization list than initialization in the constructor

More on initialization & C++11:
1-composition, 2-initialization