ОПЕРАТОР =

доц. д-р Нора Ангелова

ОПЕРАТОР =

- Оператор за присвояване
  - Извиква се при присвояване на съществуващи обекти от съответния клас (не е конструктор и не създава обект)

Пример:

```
Point2D firstPoint, secondPoint(1, 4);

// firstPoint и secondPoint са съществуващи обекти
firstPoint = secondPoint;
```

ОПЕРАТОР =

- Оператор за присвояване
  - Извиква се при присвояване на съществуващи обекти от съответния клас (не е конструктор и не създава обект)
  - Създава се по подразбиране при опит за присвояване между обекти
  - Реализира се чрез директно присвояване на член-данните

Пример:

```
Point2D firstPoint, secondPoint(1, 4);

// firstPoint и secondPoint са съществуващи обекти
firstPoint = secondPoint;
```

ОПЕРАТОР =

- Кога да дефинираме operator =?
  - Ако не е дефиниран се извършва автоматично чрез директно присвояване.

Когато не може да се извърши директно присвояване на член-данните

Пример – динамично заделяне на памет

ОПЕРАТОР =

- Специфики
  - Връща className&

Пример:

```
Point2D firstPoint, secondPoint(1, 4), thirdPoint(3, 8);
// Верижно извикване
firstPoint = secondPoint = thirdPoint;
```

Оператор =

- Специфики
  - Връща className&
  - Приема като параметър className const &

Пример:

```
Point2D& Point2D::operator=(Point2D const& pointObj) {
    // …
}
```

ОПЕРАТОР =

- Специфики
  - Връща className&
  - Приема като параметър className const &

Пример:

```
Point2D& Point2D::operator=(Point2D const& pointObj) {
    // …
    return ?!?;
}
```

ОПЕРАТОР =

- Специфики
  - Връща className&
  - Приема като параметър className const &

Пример:

```
Point2D& Point2D::operator=(Point2D const& pointObj) {
    // …
    return *this;
}
```

ОПЕРАТОР =

- Специфики
  - Връща className&
  - Приема като параметър className const &

Пример:

```
Point2D& Point2D::operator=(Point2D const& pointObj) {
    // Какво очакваме да прави?
    return *this;
}
```

ОПЕРАТОР =

- Специфики
  - Връща className&
  - Приема като параметър className const &

Пример:

```
Point2D& Point2D::operator=(Point2D const& pointObj) {
    // Вече има заделена памет за левия операнд
    // Трябва да се извърши присвояване на член-данните
    // Последното няма да работи за указатели към динамичната памет
    // Какво става при: int * = int *?!

    return *this;
}
```

ОПЕРАТОР =

- Специфики
  - Връща className&
  - Приема като параметър className const &

Пример:

```
Point2D& Point2D::operator=(Point2D const& pointObj) {
    // Вече има заделена памет за левия операнд
    // Трябва да се извърши присвояване на член-данните
    // Последното няма да работи за указатели към динамичната памет
    // Какво става при: int * = int *?! – паметта ще се подели

    return *this;
}
```

ОПЕРАТОР =

- Специфики
  - Връща className&
  - Приема като параметър className const &

Пример:

```
Point2D& Point2D::operator=(Point2D const& pointObj) {
    // Изтриване на паметта за левия операнд
    // Заделяне на нова памет и присвояване на стойностите
    return *this;
}
```

ОПЕРАТОР =

- Специфики
  - Връща className&
  - Приема като параметър className const &

Пример:

```
Point2D& Point2D::operator=(Point2D const& pointObj) {
    // Ако двата обекта съвпадат obj = obj ?!?
    // Изтриване на паметта за левия операнд ?!??
    // Заделяне на нова памет и присвояване на стойностите - кои?
    return *this;
}
```

ОПЕРАТОР =

```
<име_на_клас>& <име_на_клас>::operator= (<име_на_клас> const& obj) {
   if (this != &obj) {
     // 1. Освобождаване на динамичната памет на
     // компонентите на обекта, сочен от указателя this,
     // ако такава е отделена;
     // 2. Копиране на компонентите на obj в съответните
     // компоненти на обекта, сочен от указателя this
   }
   return *this;
}
```

ВРЕМЕ ЗА ВАШИТЕ ВЪПРОСИ

ПРИМЕР

```
class A {
  private:
    int a;
  public:
    A(int aData = 1) {
       a = aData;
       cout << "A(" << aData << ")" << endl;
    }
    A(A const & obj) {
       a = obj.a;
       cout << "Copy A(" << obj.a << ")" << endl;
    }
};

class B {
  private:
    int b;
    A objA;
  public:
    B(int bData = 1, int aData = 0) {
        b = bData;
        cout << "B(" << bData <<"," << aData << ")" << endl;
        objA = A(10);
    }
    B(B const & obj) {
       b = obj.b;
       cout << "Copy B(" << obj.b << ")" << endl;
    }
};
```

```
int main() {
   B test1(1,3);
   B test2 = test1;
   return 0;
}
```

ПРИМЕР

```
class A {
  private:
    int a;
  public:
    A(int aData = 1) {
       a = aData;
       cout << "A(" << aData << ")" << endl;
    }
    A(A const & obj) {
       a = obj.a;
       cout << "Copy A(" << obj.a << ")" << endl;
    }
};

class B {
  private:
    int b;
    A objA;
  public:
    B(int bData = 1, int aData = 0) {
        b = bData;
        cout << "B(" << bData <<"," << aData << ")" << endl;
        objA = A(10);
    }
    B(B const & obj) {
       b = obj.b;
       cout << "Copy B(" << obj.b << ")" << endl;
    }
};
```

```
int main() {
   B test1(1,3);
   B test2 = test1;
   return 0;
}
```

Резултат:

A(1)
B(1,3)
A(10)
A(1)
Copy B(1)

ПРИМЕР 2

```
class A {
  private:
    int a;
  public:
    A(int aData) {
       a = aData;
       cout << "A(" << aData << ")" << endl;
    }
    A(A const & obj) {
       a = obj.a;
       cout << "Copy A(" << obj.a << ")" << endl;
    }
};

class B {
  private:
    int b;
    A objA;
  public:
    B(int bData = 1, int aData = 0) {
        b = bData;
        cout << "B(" << bData <<"," << aData << ")" << endl;
        objA = A(10);
    }
    B(B const & obj) {
       b = obj.b;
       cout << "Copy B(" << obj.b << ")" << endl;
    }
};
```

```
int main() {
   B test1(1,3);
   B test2 = test1;
   return 0;
}
```

ПРИМЕР 2

```
class A {
  private:
    int a;
  public:
    A(int aData) {
       a = aData;
       cout << "A(" << aData << ")" << endl;
    }
    A(A const & obj) {
       a = obj.a;
       cout << "Copy A(" << obj.a << ")" << endl;
    }
};

class B {
  private:
    int b;
    A objA;
  public:
    B(int bData = 1, int aData = 0) {
        b = bData;
        cout << "B(" << bData <<"," << aData << ")" << endl;
        objA = A(10);
    }
    B(B const & obj) {
       b = obj.b;
       cout << "Copy B(" << obj.b << ")" << endl;
    }
};
```

Резултат:
Грешка

```
int main() {
    B test1(1,3);
    B test2 = test1;
    return 0;
}
```