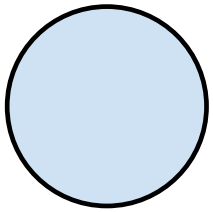
Наследяване. Базови сведения

Калин Георгиев

22 април 2016 г.

Моделиране на различни обекти с общи свойства

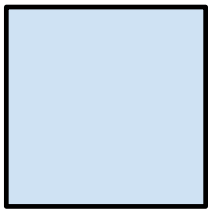
Еднакви и различни



Circle

Дефиниран чрез:
- радиус
- координати на центъра

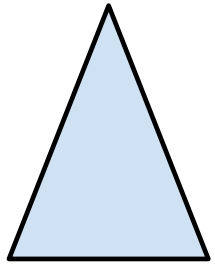
Свойства:
- периметър
- лице



Square

Дефиниран чрез:
- страна
- координати на центъра
- ъгъл с оста

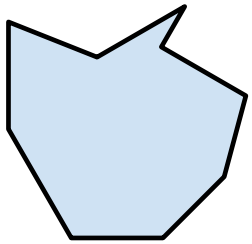
Свойства:
- периметър
- лице



Isosceles

Дефиниран чрез:
- бедро
- основа
- координати на медицентъра
- ъгъл с оста

Свойства:
- периметър
- лице



Polygon

Дефиниран чрез:
- списък координати на върхове

Свойства:
- периметър
- лице
- брой върхове
- дали е изпъкнал

Множество от различни обекти

```
int main ()
{
  Square* squares[] =
     {new Square (2,0,0,0),
      new Square (4,0,0,0),
      new Square (3,0,0,0)};

  Circle* circles[] =
     {new Circle (2,0,0),
      new Circle (4,0,0)};

  cout << sumSurf<Square> (squares,3) +
          sumSurf<Circle> (circles,2);

  return 0;
}
```

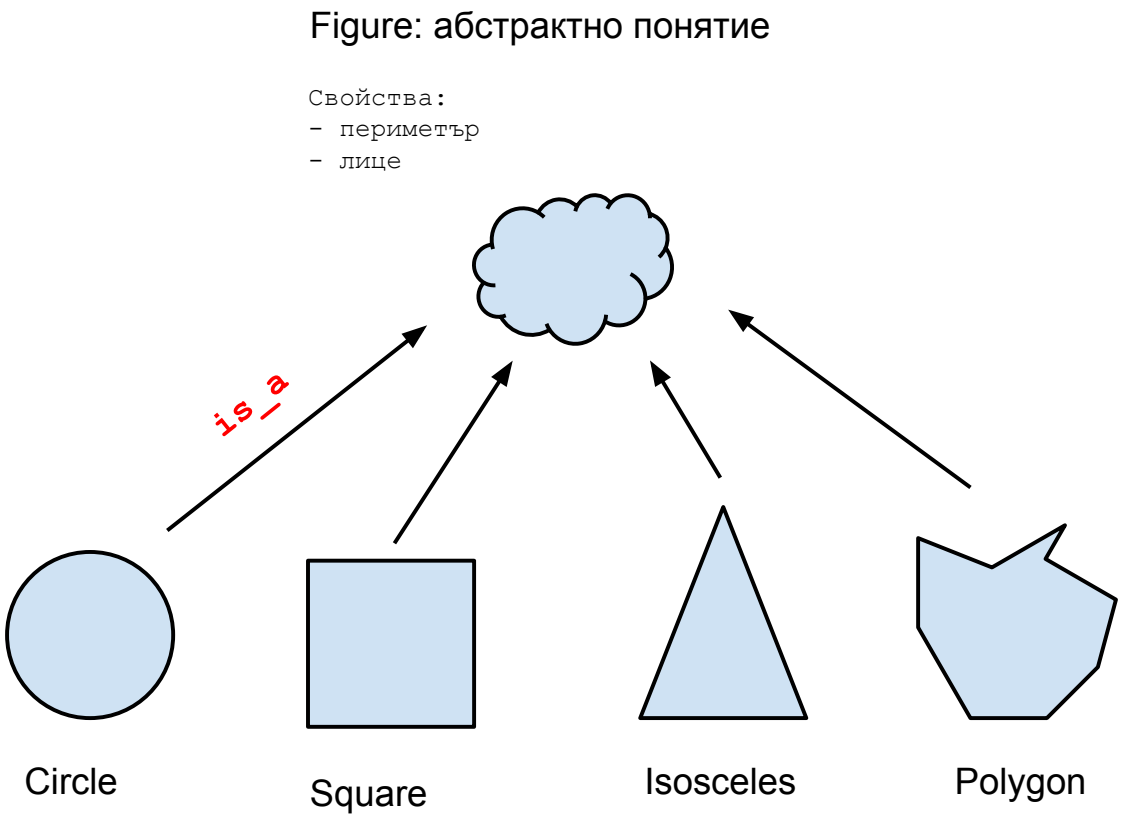
```
template <typename F>
double sumSurf (F* figures[], int n)
{
  double sum = 0;

  for (int i = 0; i < n; i++)
    sum += figures[i]->surface();

  return sum;
}
```

"Абстрахиране"от конкретния тип

Какво е фигура? Йерархия от фигури



Полиморфизъм

```
int main ()
{
  Figure* figures[] =
     {new Square (2,0,0,0),
      new Circle (2,0,0),
      new Square (4,0,0,0),
      new Square (3,0,0,0),
      new Circle (4,0,0)};

  cout << sumSurf (figures,5);

  return 0;
}
```

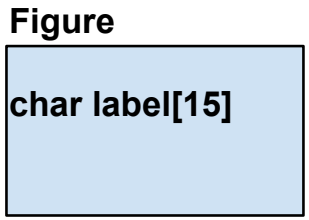
```
template <typename F>
double sumSurf (F* figures[], int n
{
  double sum = 0;

  for (int i = 0; i < n; i++)
    sum += figures[i]->surface();

  return sum;
}
```

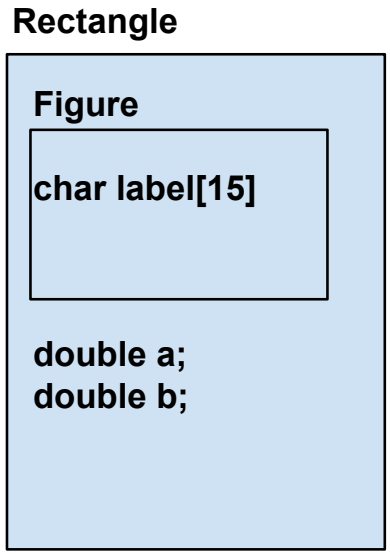
Клас Figure

```
class Figure
{
  public:
    char label[15];
    double surface ()
    {return 0;}
};
```



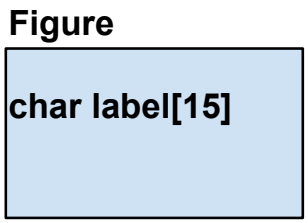
Наследяване на клас Figure

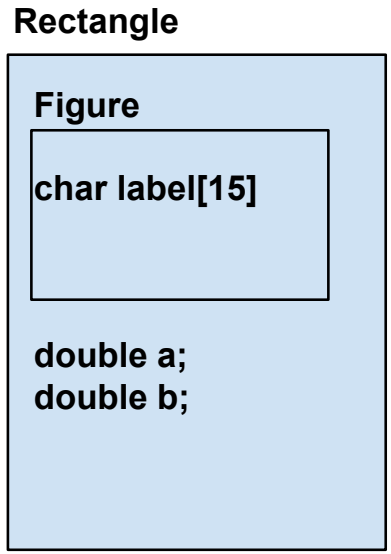
```
class Rectangle : public Figure
{
  public:
    double a,b;
    double surface ()
    {return a*b;}
};
```



Съвместимост на Figure и Rectangle

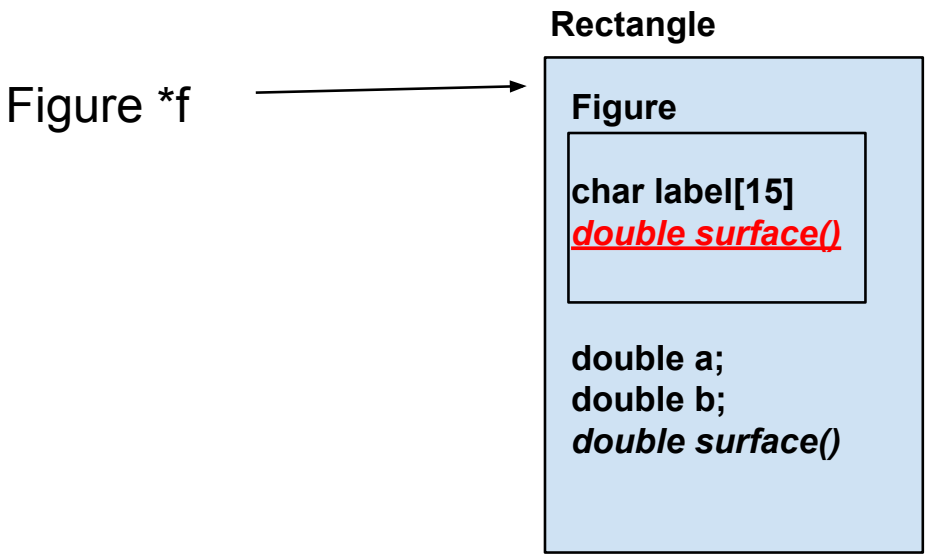
```
Rectangle r1 (2,4);
Figure f = r1;
cout << f.surface();
```





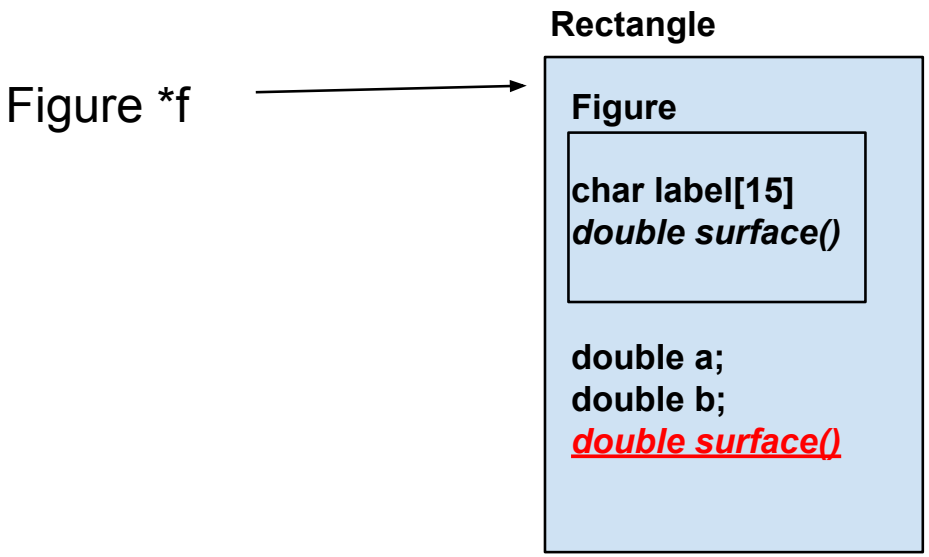
Съвместимост на указатели към Figure и Rectangle

```
Rectangle r1 (2,4);
Figure *f = &r1;
cout << f->surface();
```

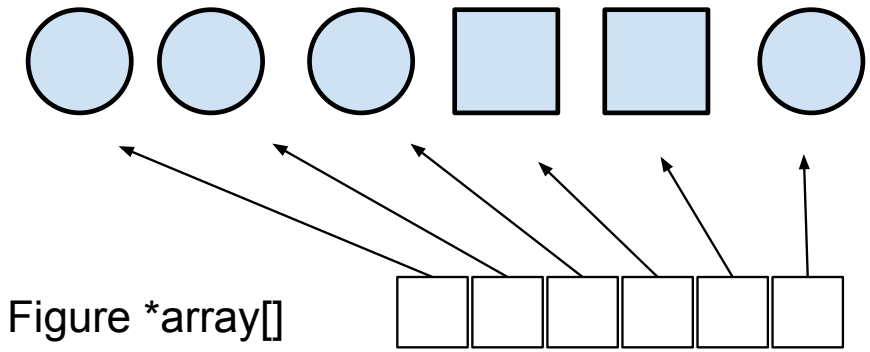


Ако Figure::surface е виртуален

```
cout << f->surface();
```



Масив от указатели към фигури



Благодаря ви за вниманието!