Contents

1 Further Properties of Inheritance

Class behaviour in inheritance
Attributes:

- descendant inherits all attributes from ancestor,
- you can add some more attributes if you need them.

Methods:

- descendant inherits all methods from ancestor,
- you can add some more methods if you need them.

1.1 Class Compatitbility

Class Compatibility

- Inheriting class has the same attributes and methods as its ancestor.
- It has the same (or extended) interface.
- The descendant can substitute its ancestor.
- The ancestor can *not* substitute its descendant.
  - Descendant can have extended interface – not all actions can be propagated to ancestor.

1.2 Methods in Inheritance

Methods in Inheritance
Descendant inherits all methods from ancestor including its implementation.

- Not all descendants need the same implementation.
- It's not always possible to define how to implement the method. We just know that it exists and that we need to use it.

2 Abstract Methods

Abstract Methods

- Every organism can reproduce itself:
  - it produces eggs, it produces grains, it bears, ...
- Every animal can find its food, ...

```
procedure CAnimal.live;
begin
  while not isDead do
  begin
    findFood;
    consumeFood;
    if canReproduce then
      reproduce;
    if needSleep then
      sleep;
  end;
end;
```

Abstract Methods II.

- ”General animal” (class CAnimal) don’t know how to find its food or how to consume it etc. Yet it can have defined (implemented) method that uses these actions.
- About ”general animal” you can say how it lives – see program on previous slide.
- But you cannot ask ”general animal” to live (to invoke the method live).
  - It doesn’t know how to find its food. Should it hunt? Should it paw?
  - It doesn’t know how to consume the food.
- Some methods of the class can have no implementation (we don’t need to know its implementation). Yet we can be certain that they exist.
- Method with no implementation is called *abstract method*.

2.1 Abstract Methods in Pascal

Abstract Methods in Pascal

- There is no support for abstract methods in Borland Pascal `:-(`
- The solution is implementation of the abstract method only with error invokation.

```
type CAbstractClass=object
       procedure abstractMethod;
     end;
procedure CAbstractClass.abstractMethod;
begin
  WriteLn('Abstract method invocation');
  halt(1);
end;
```

Abstract Methods in Pascal II.

It is possible to use Borland library support:

```
 uses Objects;
type CAbstractClass=object
       procedure abstractMethod;
     end;
procedure CAbstractClass.abstractMethod;
begin
   abstract;
end;
```

Practical Usage of Abstract Methods

- When class must provide some action but it's defined in descendants.
- Developing with *interfaces*
  - Interface is a class with only abstract methods.