Lanzamiento y propagación de excepciones

Propagación de una excepción

- Si un método que debe capturar una excepción no desea hacerlo, puede propagarla al lugar de llamada al método
- Se debe declarar la excepción en la cabecera del método con la instrucción throws:

```
metodo(){
    BufferedReader bf=new ….:
    try{
        //la llamda a readLine puede
        //provocar una IOException
        String s=bf.readLine();
    }
    catch(IOEXception ex){

    }
}
```


```
metodo() throws IOException{
    BufferedReader bf=new ….;
    //si se produce la excepción,
    //se propaga al punto de llamada
    //a metodo(), que será donde haya
    //  que capturarla
    String s=bf.readLine();
}
```

Lanzamiento de una excepción

- Desde un método de una clase se puede lanzar una excepción para que sea capturada desde el punto de llamada al método
- Para lanzar una excepción se utiliza la instrucción *throw objeto_excepcion:*

```
metodo() throws IOException{
    :
    //creación y lanzamiento de la
    //excepción
    throw new IOException();
}
```

Si se lanza una excepción checked el compilador obliga a declararla con throws para que se propague, si es RuntimeException **no es necesario declararla**

Excepciones personalizadas

- Se puede crear una excepción personalizada definiendo una clase que herede Exception:

```
class TestException extends Exception{

}


class C1{
    //propaga la excepción que lanza
    public void metodo() throws TestException{
        :
        throw new TestException();
    }
}
```


```
C1 c=new C1();
try{
      //al utilizar metodo() se debe capturar
      //la excepción
      c.metodo();
}
catch(TestException t){
    :
}
```