1. Removed the Comparable interface because it never been used. Code smell: Dead code.

**Offending Code:**

public int compareTo(Object o)

{

return 1;

}

1. Made protected field to private in the Hero class because of security.

Offending code: private double chanceToBlock;

In Class Sorceress public field changed to private to prevent access from outside class.

Offending Code:

**public** **final** **int** MIN\_ADD = 25;

**public** **final** **int** MAX\_ADD = 50;

Changed code:

**private** **final** **int** MIN\_ADD = 25;

**private** **final** **int** MAX\_ADD = 50;

In the Monster class changed two protected field to private

old code:

**protected** **double** chanceToHeal;

**protected** **int** minHeal, maxHeal;

new code :

**private** **double** chanceToHeal;

**private** **int** minHeal, maxHeal;



**Offending Code:**

Public Thief (){  
 super(“Thief”, 75, 6, .8, 20, 40, 0.5);

}

**The above code was changed to the code bellow because it more easy to read and understand.**

**private** **static** String *name*="Thief";

**private** **static** **int** *hitPoints* =75;

**private** **static** **int** *attackSpeed* =6;

**private** **static** **double** *chanceToHit*=.8;

**private** **static** **int** *damageMin*=20;

**private** **static** **int** *damageMax*=40;

**private** **static** **double** *chanceToBlock*=0.5;

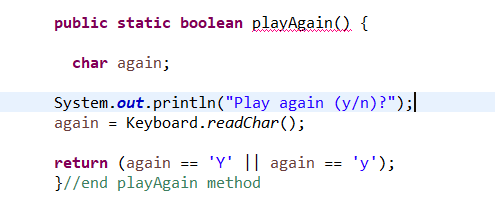
**public** Thief()

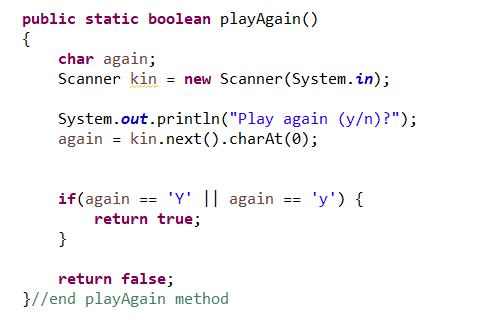
{

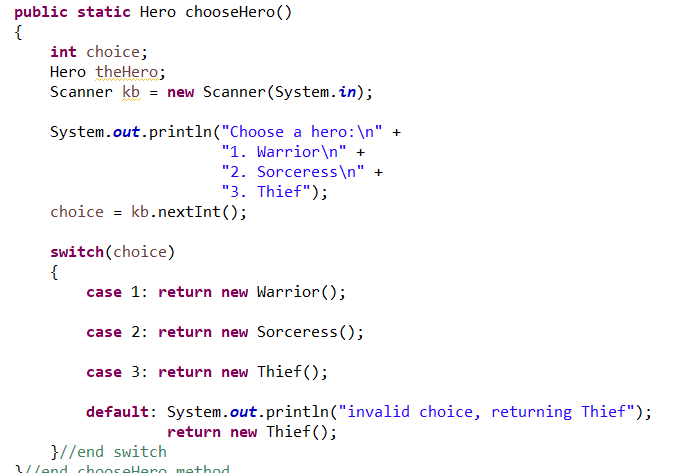
**super**(*name*, *hitPoints*, *attackSpeed*, *chanceToHit*, *damageMin*, *damageMax*, *chanceToBlock*);

}

1. Removed Keyboard class, and instituted Scanner Object throughout code

**Original:** 

**New:** 



1. We created a HeroFactory class because of object creation is done in one place.

Old code:

public static Hero chooseHero()

{

int choice;

Hero theHero;

Scanner kb = new Scanner(System.in);

System.out.println("Choose a hero:\n" +

"1. Warrior\n" +

"2. Sorceress\n" +

"3. Thief");

choice = kb.nextInt();

switch(choice)

{

case 1: return new Warrior();

case 2: return new Sorceress();

case 3: return new Thief();

default: System.out.println("invalid choice, returning Thief");

return new Thief();

}//end switch

}//end chooseHero method

**New Code :**

**package** Dungeon;

**import** java.util.Scanner;

**public** **class** HeroFactory {

Scanner input;

**public** Hero createHero(){

**int** choice;

System.*out*.println("Choose a hero:\n" +

"1. Warrior\n" +

"2. Sorceress\n" +

"3. Thief");

input = **new** Scanner(System.*in*);

choice = input.nextInt();

Hero hero = **null**;

**if**(choice == 1){

hero = **new** Warrior();

**return** hero;

}**else** **if**(choice == 2){

hero=**new** Sorceress();

**return** hero;

}**else** **if**(choice == 3){

hero = **new** Thief();

**return** hero;

}**else**{

System.*out*.println("invalid choice, returning Thief");

**return** **new** Thief();

}

}

}

**Similarly we created a MonsterFactory class**

**Old code:**

**public** **static** Monster generateMonster()

{

**int** choice;

choice = (**int**)(Math.*random*() \* 3) + 1;

**switch**(choice)

{

**case** 1: **return** **new** Ogre();

**case** 2: **return** **new** Gremlin();

**case** 3: **return** **new** Skeleton();

**default**: System.*out*.println("invalid choice, returning Skeleton");

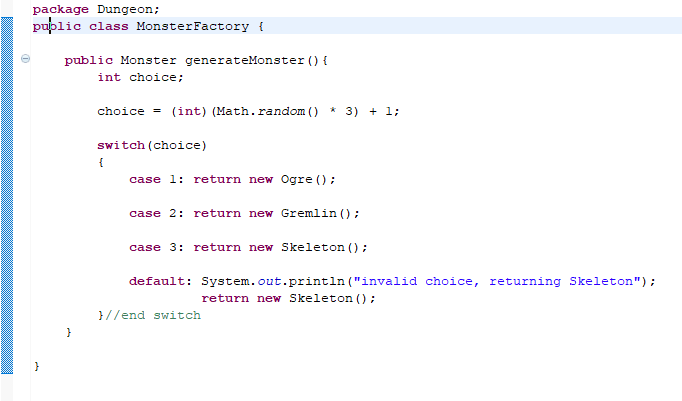
**return** **new** Skeleton();

}//end switch

}//end generateMonster method

-----------------------------------------------------

**New code:**



**In main:**

