Warrior (AWarrior)

Post-conditions of Move

A method that accepts an argument for a direction to move. The method should change Position attribute of the Warrior. If the Warrior is at Position (0,0), a move to direction SOUTH_EAST will change the Position to (1,1).

```
sut.Position.X == sut.Position.X@Pre + 1
sut.Position.Y == sut.Position.Y@Pre + 1
```

Should throw an InvalidOperationException, if a Warrior tries to move outside the Board boundary.

```
sut.Position = new Position { X = 0, Y = 0 };
Assert.That(() => sut.Move(Direction.WEST),
Throws.TypeOf<InvalidOperationException>());
```

Post-conditions of addWeapon

A method that accepts an argument for a Weapon. The method should change the boolean return type of hasWeapon() method from false to true.

```
sut.hasWeapon() == !sut.hasWeapon()@Pre
```

Should increase MeleePower if a Sword is picked and MagicPower if a Staff is picked.

```
sut.MeleePower == sut.MeleePower@Pre + weapon.Power
sut.MagicPower == sut.MagicPower@Pre + weapon.Power
```

Post-conditions of dropWeapon

The method should change the boolean return type of hasWeapon() method from true to false

```
sut.hasWeapon() == !sut.hasWeapon()@Pre
```

Post-conditions of Warrior

The constructor should set HitPoints to 100 and DefensePercentage to 0 when constructed.

```
sut.HitPoints == 100
sut. DefensePercentage == 0
```

Control Flow Testing : addWeapon method

All-Path Criteria

1-2(F)-4-5(F)-7(F)-9

Statement Coverage

1-2(F)-4-5(F)-7(F)-9
1-2(T)-3
1-2(F)-4-5(T)-6-9
1-2(F)-4-5(F)-7(T)-8-9

Example: 1-2(F)-4-5(F)-7(T)-8-9

Predicate Interpretation Table

Node	Description	Interpretation
1	<weapon>	
2(F)	this.weapon != null	
4	this.weapon = weapon	
5(F)	weapon.WeaponType == WeaponType.Sword	weapon.WeaponType == WeaponType.Sword
7(T)	weapon.WeaponType == WeaponType.Staff	weapon.WeaponType == WeaponType.Staff
8	this.MagicPower += this.weapon.Power	this.MagicPower = this.MagicPower + this.weapon.Power
9	return void	

Path Predicate Expression

```
weapon.WeaponType == WeaponType.Sword === false
weapon.WeaponType == WeaponType.Staff === true
```

Input Vector (Test Data)

```
weapon.WeaponType = WeaponType.Staff
```

Def() and c-use() Sets of Nodes

Node i	def(i)	c-use(i)
1	{weapon}	{}
2	{this.weapon}	{weapon}
3	{InvalidOperationException}	{}
4	{}	{}
5	{this.MeleePower}	{this.MeleePower, this.weapon.Power}
6	{this.MagicPower}	{this.MagicPower, this.weapon.Power}
7	{}	{}

Predicates and p-use() Set of Edges

Edges (i, j)	predicate(i, j)	p-use(i, j)
(1, 2)	~(this.weapon != null)	{this.weapon}
(1, 3)	(this.weapon != null)	{this.weapon}
(2, 4)	~(weapon.WeaponType == WeaponType.Sword)	{weapon.WeaponType}
(2, 5)	(weapon.WeaponType == WeaponType.Sword)	{weapon.WeaponType}
(4, 6)	(weapon.WeaponType == WeaponType.Staff)	{weaon.WeaponType}
(4,7)	~(weapon.WeaponType ==	{weapon.WeaponType}

	WeaponType.Staff)	
(5,7)	True	{}
(6,7)	True	{}

All-defs on weapon

Global definitions: 1,2

Global c-use: 2,5,6

Def-clear path: 2-5

Complete path: 1-2-5-7

P-uses: (1,2), (1,3), (2,4), (2,5), (4,6), (4,7)

Def-clear path: 2-4

Complete path: 1-2-4-7

Weapon (AWeapon)

Post-conditions of Weapon

A constructor that accepts an argument for WeaponType (Sword, Staff). It should set the Power a random number between 3 and 6 for Staff WeaponType and a random number between 2 and 4 for Sword WeaponType.

Sword (WeaponType)

```
sut.Power >= 3
sut.Power <= 6
```

Staff (WeaponType)

```
sut.Power >= 2
sut.Power <= 4
```

Control Flow Testing : Weapon constructor

All-Path Criteria

1-2-3(T)-4

Statement Coverage

1-2-3(T)-4
1-2-3(F)-4

Example: 1-2-3(T)-4

Predicate Interpretation Table

Node	Description	Interpretation
1	<weaponType>	
2	this.weaponType = weaponType	this.weaponType = weaponType
3(T)	this.weaponType == WeaponType.Staff	this.weaponType == WeaponType.Staff
4	this.Power = Util.random.Next(3,7)	this.Power >= 3 and < 7
5	this.Power = Util.random.Next(2,5)	this.Power >= 2 and < 5

Path Predicate Expression

```
this.weaponType == WeaponType.Staff === true
```

Input Vector (Test Data)

```
weaponType = weaponType.Staff
```

Def() and c-use() Sets of Nodes

Node i	def(i)	c-use(i)
1	{weaponType}	{}
2	{this.weaponType}	{weaponType}
3	{this.Power}	{}
4	{this.Power}	{}
5	{}	{}

Predicates and p-use() Set of Edges

Edges (i, j)	predicate(i, j)	p-use(i, j)
(1, 2)	True	{}
(2, 3)	((this.weaponType == WeaponType.Staff))	{weaponType}
(2, 4)	~((this.weaponType == WeaponType.Staff))	{}
(3, 5)	True	{}
(4, 5)	True	{}

All-defs on weaponType

Global definitions: 1

Global c-use: 2

Def-clear path: 1-2

Complete path: 1-2-4-5

P-uses: (2,3)

Def-clear path: 1-2-3

Complete path: 1-2-3-5

MeleeWarrior (AMeleeWarrior)

Post-conditions of MeleeWarrior

A constructor that should set the MeleePower a random number between 5 and 10 and MagicPower a random number between 1 and 3.

```
sut.MeleePower >= 5
sut.MeleePower <= 10

sut.MagicPower >= 1
sut.MagicPower <= 3
```

MagicWarrior (AMagicWarrior)

Post-conditions of MagicWarrior

A constructor that should set the MeleePower a random number between 1 and 3 and MagicPower a random number between 3 and 8.

```
sut.MeleePower >= 1
sut.MeleePower <= 3

sut.MagicPower >= 3
sut.MagicPower <= 8
```