Heroes in Training 

Let's build the beginning of a small RPG character system.

We're going to model heroes that can train and grow stronger.

Your Task

1. Create an abstract class called `Hero`

Add two properties:

- `Name` (string)
- `Level` (int)

Add:

- A constructor that sets `Name`
- `Level` should default to `1`

Add an abstract method:

```
public abstract string Describe();
```

This forces all hero types to describe themselves differently.

2. Create an interface called `ITrainable`

It should contain one method:

```
void Train();
```

Anything that is *trainable* can increase in level.

3. Create a class called `Mage`

- Inherit from `Hero`
- Implement `ITrainable`
- Add a property: `Mana` (int)

Requirements:

- Default `Mana` to 30
- Override `Describe()`
- Implement `Train()` so that:

    - `Level++`
    - `Mana += 5`

4. Create a class called `Warrior`

- Inherit from `Hero`
- Implement `ITrainable`
- Add a property: `Stamina` (int)

Requirements:

- Default `Stamina` to 80
- Override `Describe()`
- Implement `Train()` so that:
    - `Level++`
    - `Stamina += 10`

Expected Output (Example)

When running the application, the console should resemble the following outputs:

```
Before training:

Mage: Nia (Level 1) - Mana: 30
Warrior: Koba (Level 1) - Stamina: 80

Training...

After training:

Mage: Nia (Level 2) - Mana: 35
Warrior: Koba (Level 2) - Stamina: 90
```

Hint: Store a List of Heroes to call `Describe()`. Then when you want to look to `Train()`, you can use a new keyword:

```
foreach(var hero in heroes)
{
    if(hero is ITrainable) // Checks if the Hero also is a ITrainable. Meaning, it
can be trained.
    {
        hero.Train();
    }
}
```

Reminders

- `Hero` must be `abstract`
- `Describe()` must be overridden in both child classes
- Use `: Hero, ITrainable` when inheriting and implementing
- `Level++` happens inside `Train()`

Bonus (Optional)

1. Add validation so:
    - `Name` cannot be empty
    - `Level` cannot be less than 1
2. Create a new hero type:
    - `Archer`
    - Give it its own stat (e.g. `Accuracy`)
    - Make it trainable