এই প্রজেক্টটি একটি **ভেহিকল সিমুলেশন সিস্টেম** তৈরি করার জন্য জাভা প্রোগ্রামিং ভাষায় লেখা হয়েছে। এটি ইন্টারফেস, ইন্টারফেস এক্সটেনশন, কনস্ট্যান্ট, `instanceof` চেক এবং সিলড ইন্টারফেসের ব্যবহার বোঝার জন্য ডিজাইন করা হয়েছে। নিচে প্রজেক্টটির বিস্তারিত বর্ণনা বাংলায় দেওয়া হলো:

প্রজেক্টের উদ্দেশ্য

- **ইন্টারফেস বোঝা**: কীভাবে ইন্টারফেস ব্যবহার করে বিভিন্ন গাড়ির শেয়ারড আচরণ (shared behavior) সংজ্ঞায়িত করা যায়।
- **ইন্টারফেস এক্সটেনশন**: একটি ইন্টারফেস থেকে আরেকটি ইন্টারফেস তৈরি করা এবং এর মাধ্যমে অতিরিক্ত ফাংশনালিটি যোগ করা।
- **কনস্ট্যান্ট ব্যবহার**: ইন্টারফেসে কনস্ট্যান্ট ব্যবহার করে সর্বোচ্চ গতির সীমা নির্ধারণ।
- `instanceof` **চেক**: কোনো অবজেক্ট কোন ইন্টারফেসের ইনস্ট্যান্স কিনা তা পরীক্ষা করা।
- **সিলড ইন্টারফেস**: নির্দিষ্ট ক্লাস ছাড়া অন্য কোনো ক্লাসকে ইন্টারফেস ইমপ্লিমেন্ট করতে বাধা দেওয়া।

প্রজেক্টের উপাদান

প্রজেক্টটি কয়েকটি ফাইলে বিভক্ত, যেখানে প্রতিটি ফাইল একটি নির্দিষ্ট ক্লাস বা ইন্টারফেস ধারণ করে।

1. Moveable.java

- **বর্ণনা**: এটি একটি ইন্টারফেস যা সকল গাড়ির জন্য সাধারণ আচরণ সংজ্ঞায়িত করে, যেমন একটি নির্দিষ্ট অবস্থানে (x, y) সরানো।
- **মূল উপাদান**:
  - `move(double x, double y)` : এই মেথডটি গাড়িকে একটি নতুন অবস্থানে নিয়ে যায়।
  - এটি সকল গাড়ির জন্য সাধারণ, যেমন গাড়ি, সাইকেল এবং ইলেকট্রিক স্কুটার।

2. Powered.java

- **বর্ণনা**: এটি `Moveable` ইন্টারফেসকে প্রসারিত করে এবং শক্তিচালিত গাড়ির জন্য অতিরিক্ত ফাংশনালিটি যোগ করে। এটি একটি সিলড ইন্টারফেস, যা শুধুমাত্র `Car` এবং `ElectricScooter` ক্লাস দ্বারা ইমপ্লিমেন্ট করা যায়।
- **মূল উপাদান**:
  - `SPEED_LIMIT` : একটি কনস্ট্যান্ট (120 কিমি/ঘণ্টা) যা সর্বোচ্চ গতির সীমা নির্ধারণ করে।
  - `milesPerGallon()` : এই মেথডটি জ্বালানি দক্ষতা (মাইল প্রতি গ্যালন) ফেরত দেয়।
  - `permits Car, ElectricScooter` : নিশ্চিত করে যে শুধুমাত্র এই দুটি ক্লাস এই ইন্টারফেস ব্যবহার করতে পারে।

3. Car.java

- **বর্ণনা**: এই ক্লাসটি `Powered` ইন্টারফেস ইমপ্লিমেন্ট করে, যা একটি জ্বালানি চালিত গাড়ির প্রতিনিধিত্ব করে।
- **মূল উপাদান**:
  - **ফিল্ডস**: `x` , `y` (অবস্থান), `speed` (গতি), এবং `fuelEfficiency` (জ্বালানি দক্ষতা)।
  - **কনস্ট্রাক্টর**: জ্বালানি দক্ষতা দিয়ে গাড়ি ইনিশিয়ালাইজ করে।
  - `move()` : গাড়িকে নতুন অবস্থানে নিয়ে যায়, তবে গতি `SPEED_LIMIT` এর মধ্যে থাকতে হবে।
  - `milesPerGallon()` : গাড়ির জ্বালানি দক্ষতা ফেরত দেয়।
  - `setSpeed()` : গাড়ির গতি সেট করে।

4. Bicycle.java

- **বর্ণনা**: এই ক্লাসটি শুধুমাত্র `Moveable` ইন্টারফেস ইমপ্লিমেন্ট করে, কারণ এটি একটি অ-শক্তিচালিত গাড়ি।
- **মূল উপাদান**:
  - **ফিল্ডস**: `x` , `y` (অবস্থান)।
  - **কনস্ট্রাক্টর**: সাইকেলের অবস্থান শূন্যে ইনিশিয়ালাইজ করে।
  - `move()` : সাইকেলকে নতুন অবস্থানে নিয়ে যায়, কোনো গতির সীমা ছাড়াই।

5. ElectricScooter.java

- **বর্ণনা**: এই ক্লাসটি `Powered` ইন্টারফেস ইমপ্লিমেন্ট করে এবং একটি ইলেকট্রিক স্কুটারের প্রতিনিধিত্ব করে। এটি `final` হিসেবে চিহ্নিত, যাতে এটি থেকে কোনো সাবক্লাস তৈরি করা না যায়।
- **মূল উপাদান**:
  - **ফিল্ডস**: `x`, `y` (অবস্থান), `speed` (গতি), এবং `batteryEfficiency` (ব্যাটারি দক্ষতা)।
  - **কনস্ট্রাক্টর**: ব্যাটারি দক্ষতা দিয়ে স্কুটার ইনিশিয়ালাইজ করে।
  - `move()`: স্কুটারকে নতুন অবস্থানে নিয়ে যায়, গতি `SPEED_LIMIT` এর মধ্যে থাকতে হবে।
  - `milesPerGallon()`: ব্যাটারির সমতুল্য জ্বালানি দক্ষতা ফেরত দেয়।
  - `setSpeed()`: স্কুটারের গতি সেট করে।

6. VehicleSimulation.java

- **বর্ণনা**: এটি প্রধান ক্লাস যা সিমুলেশন চালায় এবং সমস্ত ক্লাসের কার্যকারিতা পরীক্ষা করে।
- **মূল উপাদান**:
  - **মেইন মেথড**: তিনটি গাড়ির অবজেক্ট তৈরি করে: গাড়ি, সাইকেল এবং ইলেকট্রিক স্কুটার।
  - `instanceof` **চেক**: প্রতিটি অবজেক্ট `Moveable` এবং `Powered` ইন্টারফেসের ইনস্ট্যান্স কিনা তা পরীক্ষা করে।
  - **মুভমেন্ট টেস্ট**: গাড়ি এবং স্কুটারের জন্য গতির সীমা পরীক্ষা করে এবং সাইকেলের জন্য সাধারণ মুভমেন্ট পরীক্ষা করে।
  - **জ্বালানি দক্ষতা টেস্ট**: শক্তিচালিত গাড়ির জন্য জ্বালানি দক্ষতা প্রদর্শন করে।

প্রজেক্টের আউটপুট

প্রোগ্রামটি চালালে নিম্নলিখিত ধরনের আউটপুট দেখা যাবে:

- `instanceof` চেক দেখাবে কোন গাড়ি কোন ইন্টারফেসের সাথে সম্পর্কিত।
- গাড়ি এবং স্কুটারের জন্য গতির সীমা পরীক্ষা করা হবে। যদি গতি 120 কিমি/ঘণ্টার বেশি হয়, তবে একটি সতর্কতা বার্তা দেখাবে।
- জ্বালানি দক্ষতার মান প্রদর্শিত হবে (যেমন, গাড়ির জন্য 25 MPG, স্কুটারের জন্য 50 MPG)।

শেখার বিষয়

- **ইন্টারফেস**: কীভাবে ইন্টারফেস সাধারণ আচরণ সংজ্ঞায়িত করে।
- **ইন্টারফেস এক্সটেনশন**: `Powered` ইন্টারফেস কীভাবে `Moveable` থেকে প্রসারিত হয়।
- **কনস্ট্যান্ট**: `SPEED_LIMIT` কীভাবে সর্বোচ্চ গতি নিয়ন্ত্রণ করে।
- **সিলড ইন্টারফেস**: কীভাবে নির্দিষ্ট ক্লাস ছাড়া অন্য ক্লাসকে ইমপ্লিমেন্টেশন থেকে বিরত রাখা যায়।
- `instanceof`: কীভাবে অবজেক্টের টাইপ চেক করা যায় এবং নিরাপদে কাস্ট করা যায়।

এই প্রজেক্টটি জাভার উন্নত ফিচারগুলো বোঝার জন্য একটি ব্যবহারিক উদাহরণ প্রদান করে এবং বাস্তব জগতের সিমুলেশন তৈরির জন্য একটি শক্ত ভিত্তি তৈরি করে।