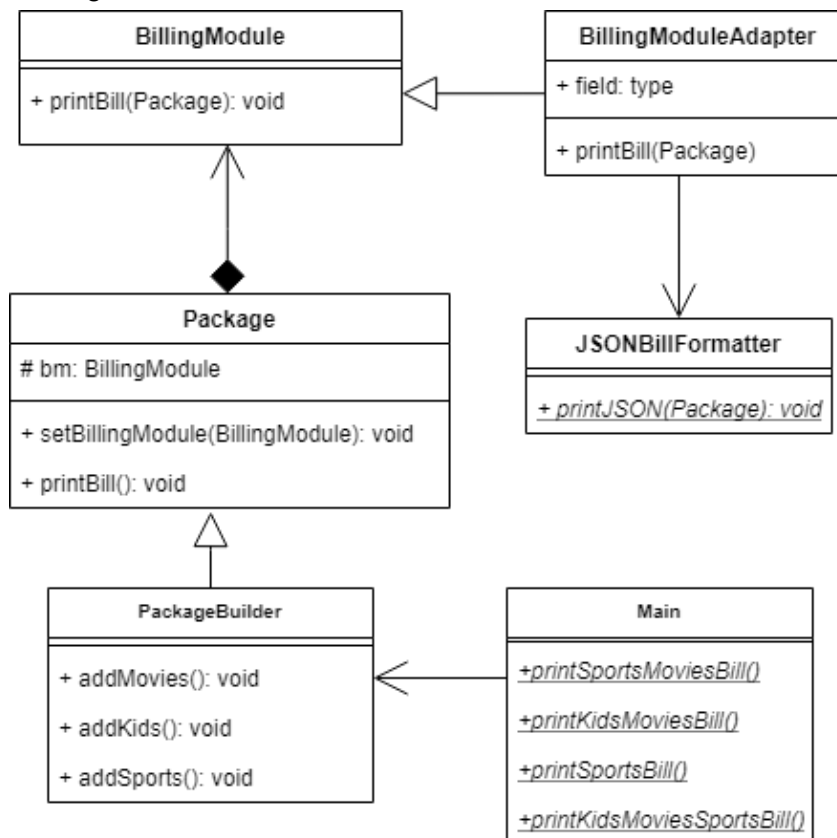# Sample Solution

A. A subclass must be made for every bundle, this can cause the codebase to grow rapidly as bundles can be made of combinations of other bundles.

   Dependency Inversion Violation, main to dependent on too many low level classes.

   Would not entertain SRP violation for main, its just interested in printing any possible package.

   JSONBillFormatter must be integrated with the system so the bills are printed as JSON in addition to its default format.

B. I would apply the builder pattern to eliminate the need to create several subclasses. Secondly, I will use the adapter pattern so that the adapter can be attached to the Package to add the JSON functionality to printBill() without changing the BillingModule or Package class.

```
┌─────────────────────────────┐          ┌─────────────────────────────┐
│        BillingModule        │          │     BillingModuleAdapter    │
├─────────────────────────────┤◁─────────├─────────────────────────────┤
│ + printBill(Package): void  │          │ + field: type               │
└─────────────────────────────┘          ├─────────────────────────────┤
              △                          │ + printBill(Package)        │
              ◆                          └─────────────────────────────┘
┌─────────────────────────────┐                        │
│           Package           │                        ▽
├─────────────────────────────┤          ┌─────────────────────────────┐
│ # bm: BillingModule         │          │       JSONBillFormatter     │
├─────────────────────────────┤          ├─────────────────────────────┤
│ + setBillingModule(BillingModule): void│ + printJSON(Package): void  │
│ + printBill(): void         │          └─────────────────────────────┘
└─────────────────────────────┘
              △
┌─────────────────────────────┐          ┌─────────────────────────────┐
│       PackageBuilder        │          │            Main             │
├─────────────────────────────┤          ├─────────────────────────────┤
│ + addMovies(): void         │◁─────────│ +printSportsMoviesBill()    │
│ + addKids(): void           │          │ +printKidsMoviesBill()      │
│ + addSports(): void         │          │ +printSportsBill()          │
└─────────────────────────────┘          │ +printKidsMoviesSportsBill()│
                                         └─────────────────────────────┘
```

C. [Solution on Repl](#)

Focus on how the builder is used in main, that it decouples main from numerous child classes. Show how the numerous child classes are eliminated