Chce przedstawić klasę Mine i TeamSplitter. W klasie Mine była realizowana praca przez krasnoludy. Główna metoda PerformMining wykorzystuje listę obiektów, które implementują interfejs IWork. Ten interfejs jest odpowiedzialny za wykonywanie pracy przez krasnoludy, mogę one wykonywać tą sama profesje na różne sposoby, ponieważ istnieją różne strategie pracy. Natomiast interfejs IOutputWriter jest związany z loggerem, którego zadaniem jest wyświetlanie danych na ekranie konsoli. Moim zdaniem ta struktura jest elastyczna i łatwo testowalna. Dowodem jest test, który dołączyłem. Używam tam mockow co daje mi przewidywane rezultaty przez co mogę testować elementy zależne i randomowe. W klasie TeamSplitter zostało zrealizowane wydzielanie pojedynczych podzespołów/podlist krasnoludów z głównej listy. Rozwiązanie, które zastosowałem jest proste, ale za to elastyczne jeszcze łatwiejsze do przetestowania i skuteczne. Kod jest przejrzysty i prosty to interpretacji.

Błędy jakie zauważam, to niespójność nazewnictwa dla tych samych bytów w obrębie kilku klas. Przykładem jest List<IWork> gdzie w klasie Mine nazywa się „allWorkers”, a w klasie TeamSplitter „workers”. Ten problem można wyeliminować przez ustalenie jednego wspólnego nazewnictwa co poprawi zwięzłość i logikę kodu.







