**Autorzy:**

**Projekt PROI: Firma testująca gry komputerowe**

**Paweł Kochański**

**Jakub Proboszcz**

**1. Założenia projektu**

**2. Hierarchia klas**

**3. Opis działania symulacji**

**4. Wybrane elementy biblioteki STL**

**Elementy używane do obsługi strumieni i komunikacji z użytkownikiem:**

- **cerr** – wyjście standardowe błędu użyte do wypisania informacji o błedach w funkcji main

- **cout** – wyjście standardowe użyt do wypisywania informacji

- **endl** – manipulator przejścia do nowej linii

- **ostream** – do przeładowywania operatorów <<

- **stringstream** – do symulacji wyjścia na potrzeby testów

- **this\_thread::sleep\_for** – odczekanie czasu po wypisaniu wiadomości

**Elementy użyte do edycji zmiennych tekstowych (string):**

- **string** – podstawowy typ zmiennej składającej się z wielu znaków

- **stoi** – konwersja ciągu znaków na liczbę całkowitą

- **to\_string** – konwersja zmiennej na ciąg znaków

- **string::npos** – pokazuje że nie znaleziono danych znaków w danym ciągu znaków

**Klasy bazowe wyjątków:**

- **invalid\_argument** – niepoprawny argument

- **out\_of\_range** – poza zasięgiem

- **logic\_error** – błąd logiczny

- **exception** – jakikolwiek wyjątek

**Kolekcje obiektów:**

- **vector** – kolekcja przyległych elementów w postaci tablicy o zmiennym rozmiarze

- **queue** – kolekcja działająca na zasadzie (first in – first out)

- **deque** – kolekcja pozwalające na łatwe dodanie i usunięcie elementów z początku i końca

- **list** – kolekcja umożliwiająca dodawanie i usuwanie elementów w dowolnym miejscu

- **find** – znajdowanie elementu w danej kolekcji

- **find\_if** – znajdowanie elementu spełniającego dany warunek w danej kolekcji

**Generowanie liczb losowych:**

- **chrono::system\_clock::now().time\_since\_epoch().count()** – odliczenie czasu od 1 stycznia 1970 w celu wygenerowania seed’a do generatorów liczb losowych

- **mt19937** – wybrany generator liczb losowych

**„Inteligentne” wskaźniki:**

- **shared\_ptr** – „inteligentny” wskaźnik do danego obiektu mogący występować w wielu miejscach w pamięci

- **make\_shared** – tworzy shared\_ptr do danego obiektu

- **unique\_ptr** – „inteligentny” wskaźnik do danego obiektu mogący występować tylko w jednym miejscu w pamięci

- **make\_unique** – tworzy unique\_ptr do danego obiektu

- **move** – przesuwanie unikatowych wskaźników na inne miejsce w pamięci

**Zarządzanie plikami:**

- **ofstream** – strumień pliku wyjściowego

- **ifstream** – strumień pliku wejściowego

- **getline** – zapisanie linii z pliku wejściowego do zmiennej w programie

**5. Zdefiniowane sytuacje wyjątkowe i ich obsługa**

W programie zostały zdefiniowane następujące sytuacje wyjątkowe:

- **ConversionError**

- niepoprawna próba skonwertowania stringa na dany typ

- **DuplicateGameError**

- próba dodania do bazy danych producenta gry, która już się tam znajduje

- **EmptyNameException**

- próba ustawienia nazwy na składającą się wyłącznie z whitespace’ów

- **EndOfFileError**

- próba zczytania danych z pliku po dojściu do jego końca

- **FileError**

- błąd otwarcia pliku

- **GameAlreadyTestedError**

- próba rozpoczęcia testowania przetestowanej gry

- **GameNotPresentError**

- próba zmiany gry w bazie danych producenta, kiedy ona się tam nie znajduje

- **GameNotRequestedError**

- próba rozpoczęcia testowania gry bez uprzedniej prośby o to

- **IncorrectAddressException**

- próba utworzenia adresu z kodem pocztowym w formacie innym niż dd-ddd (d - cyfra)

- **InvalidFileSize**

- próba ustawienia rozmiaru plików gry na 0

- **InvalidFullLength**

- próba ustawienia pełnej długości gry na krótszą od bazowej długości gry

- **InvalidId**

- próba ustawienia id objektu na znajdujący się poza jego zakresem dozwolonych id

- **InvalidLength**

- próba ustawienia długości gry na 0

- **InvalidPrice**

- próba ustawienia ceny na nieodpowiednią wartość (ujemna wartość, ujemna liczba groszy, liczba groszy przekraczająca 100)

- **InvalidProducer**

- próba dodania gry wyprodukowanej przez innego producenta do bazy danych producenta

- **InvalidTestersAmount**

- próba ustawienia minimalnej liczby testerów wymaganej do przetestowania danej gry na 0, bądź próba ustawienia maksymalnej liczby testerów na mniejszą od minimalnej

- **InvalidTitle**

- próba ustawienia tytułu gry na składający się wyłącznie z whitespace’ów

- **NoGamesUntestedError**

- próba pobrania gry do testów, gdy żadna nie jest dostępna.

- **TestingEndedError**

- próba modyfikacji rekordu przetestowanej gry

- **TestingNotEndedError**

- próba ustawienia końcowych parametrów nie w pełni przetestowanej gry

- **ZeroException**

- próba ustawienia na zero wartości, która nie może być zerem.

**Obsługa sytuacji wyjątkowych**:

Wyjątki są łapane w funkcji przeprowadzającej symulację, wypisywane do pliku i na wyjście standardowe cout po czym zgłaszane jest **ShutdownException** sygnalizujące ową sytuację wyjątkową dla programu i kończące tym samym symulację wypisując informację o tym na wyjście standardowe cerr.

Poza tym **EndOfFileError** jest wykorzystywany przy tworzeniu producentów by zakończyć ich tworzenie bez potrzeby informowania programu o ich ilości.

**6. Podział obowiązków**

**Paweł Kochański:**

Klasy:

- Address

- Worker

- Manager

- Tester

- TestingCompany

- InputFileHandler

- OutputHandler

- Simulation

Interfejs AbstractWorker

Wyjątki związane z ww. klasami

Testy jednostkowe ww. klas

**Jakub Proboszcz:**

Klasy:

-Price

- Game

- Puzzle

- CompetitiveGame

- InfiniteGame

- RoleplayingGame

- Producer

- ProducerDatabase

- TestingDatabase

- TestingRecord

Interfejs AbstractGame

Wyjątki związane z ww. klasami

Testy jednostkowe ww. klas

**Wspólnie:**

- poprawianie zauważonych drobnych bugów w plikach drugiej osoby

- main.cpp

- sprawozdanie

- testowe pliki wejściowe

- debugowanie i testy integracyjne