Akademia Górniczo-Hutnicza
im. Stanisława Staszica w Krakowie

Wydział Informatyki, Elektroniki i Telekomunikacji

Recenzja projektu „Implementacja kopca oraz kolejki”

Autorzy projektu: Magdalena Cudak, Julia Dutkiewicz

Autorzy recenzji: Łukasz Stanik, Andrzej Szaflarski

Kraków, 2018

1. Wstęp:

Projekt stworzyły Magdalena Cudak oraz Julia Dutkiewicz. Jego celem było zaimplementowanie kopca(pairing heap), kolejki(queue) oraz kolejki podwójnej (dequeue). Dodatkowo podjęto się przetestowania prawidłowości działania struktur oraz zapisania dokumentacji.

2. Organizacja kodu:

Kod projektu został podzielony na trzy katalogi: z właściwym kodem źródłowym ‘src’, z testami ‘test’ oraz katalog ‘app’ prezentujący działanie funkcji.
Kod źródłowy został podzielony na trzy moduły: Dequeue.hs, Heap.hs oraz Queue.hs. Taki podział wydaje się być prawidłowym, gdyż jest intuicyjny i spodziewany. Każdy z trzech modułów ma podobną strukturę. Najpierw następuje przedstawienie nazw dostępnych funkcji wraz z typami przyjmowanymi i zwracanymi, kolejno widzimy interfejs, który opisuje działanie każdej z funkcji, a następnie przedstawiona jest właściwa implementacja poszczególnych struktur. Poszczególne moduły implementują funkcje, których można spodziewać się po realizowanych strukturach danych. Ważnym jest, że dla każdej struktury został zdefiniowany i nazwany typ (za pomocą instrukcji ‘data’), który ją reprezentuje.

3. Czytelność projektu oraz kodu:

Moduły zostały prawidłowo podzielone na trzy katalogi w sposób standardowy. Generalnie kod źródłowy został napisany w sposób czytelny dla użytkownika. Można zauważyć identyczną kolejność elementów każdego modułu, co ułatwia i przyspiesza szukanie pożądanych informacji. Wcięcia oraz odstępy pomiędzy liniami w większości zostały estetycznie wykonane. Ważnym jest stosowanie nazw funkcji, które są intuicyjne i zgodne z nieformalnym standardem dla każdej z struktur danych.
Czepiając się nieco szczegółów, można jednak zauważyć, że brak jest pewnej konsekwencji w początkowym przedstawieniu dostępnych funkcji. Mianowicie w module Queue.hs oraz Dequeue.hs zauważamy komentarze z typami przyjmowanymi i zwracanymi, natomiast w module Heap.hs takich komentarzy brak. Co prawda typy przedstawione są prawidłowo w interfejsie, jednak oczekiwanym jest stosowanie jednej konwencji we wszystkich trzech modułach.

Dodatkowo, w module Queue.hs w interfejsie napotykamy najpierw funkcje linia po linii, jedna pod drugą, a następnie z przerwami na białe wiersze. Pogarsza to nieco czytelność kodu.
Mimo kilku uwag, trzeba stwierdzić, że kod jest zdecydowanie napisany w sposób przejrzysty, choć miejscami estetyka nie jest dopieszczona.

4. Testy

Katalog z testami został podzielony na moduły, odpowiadające za testy jednostkowe dla poszczególnych struktur, testy korzystające z oprogramowania QuickCheck oraz moduł zawierający wszystkie napisane testy. Pomysł z modułem, który gromadzi wszystkie testy wydaje się trafny i ułatwia całościowe przeglądanie projektu. Na uwagę zasługuje bardzo duża ilość testów jednostkowych wykorzystujących HUnit (łącznie zapisano 270 linii kodu odpowiedzialnego za testowanie).
Zaletą jest pamiętanie o testowaniu z wykorzystaniem QuickCheck, które skłania do przeanalizowania i dogłębnego zrozumienia działania kodu. W ten sposób przetestowano zarówno obie kolejki, jak i kopiec. Tego rodzaju napisanych testów znajdujemy niewiele, bo tylko 4, ale te które są, zaimplementowane są prawidłowo.

5. Komentarze oraz dokumentacja:

Napisane funkcje zostały opatrzone stosownymi komentarzami. Dzięki ich stosowaniu kod jest zrozumiały. Brak jest komentarzy przy funkcjach testujących, jednak odpowiednie napisy wewnątrz poszczególnych funkcji sprawiają, że takowe komentarze byłyby zbędne. Dodatkowo napisano plik package.yaml, na podstawie którego w oprogramowaniu Cabal wygenerowano plik, który w standardowy sposób prezentuje strukturę projektu.

6. Podsumowanie:

Naszym zdaniem projekt został zrealizowany bardzo dobrze. Funkcje na kolejkach oraz kopcu były kompletne i pozwalały w pełni korzystać ze struktur. Wszystkie zakładane funkcjonalności zostały przetestowane i testy owe zdały. Na uwagę zasługuje plik Main.hs, dzięki któremu nawet osoba nie znająca haskella, a mająca jedynie podstawowe wiadomości o strukturach danych może zobaczyć, jak działają poszczególne struktury danych bez wgłębianie się w szczegóły implementacyjne.