AKADEMIA GÓRNICZO-HUTNICZA

Dokumentacja do projektu

Biblioteka do obsługi stosu

z przedmiotu

Języki Programowania Obiektowego

Elektronika i Telekomunikacja 3 rok

Maciej Filipiak

grupa piątek 9:45

prowadzący: mgr inż. Jakub Zimnol

10.01.2025

1. Cel projektu

Celem projektu było stworzenie biblioteki do obsługi stosu w języku C++, która umożliwia dynamiczne dodawanie i usuwanie elementów ze stosu. Biblioteka oferuje również funkcjonalność czyszczenia stosu, wypisywania elementów znajdujących się na stosie oraz zwracania liczby elementów na stosie. Działanie stosu zostało zaprezentowane za pomocą interaktywnego menu w aplikacji, co pozwala użytkownikowi na zarządzanie stosami w czasie rzeczywistym.

2. Opis wykonania

Klasa mf::Stack

Klasa Stack jest główną klasą projektu, implementującą funkcjonalność stosu. Umożliwia dynamiczne zarządzanie elementami stosu, a także jego kontrolę w czasie rzeczywistym.

Pola:

- **int* data**
  Wskaźnik na dynamicznie alokowaną tablicę, przechowującą elementy stosu.
- **int capacity**
  Maksymalna liczba elementów, które mogą być przechowywane na stosie.
- **int topIndex**
  Indeks wskazujący na wierzchołek stosu. Wynosi -1, gdy stos jest pusty.

Konstruktor:

- **Konstruktor domyślny (Stack())**
  Inicjalizuje stos z domyślną pojemnością wynoszącą 10 elementów.
- **Konstruktor parametryczny (Stack(int maxCapacity))**
  Umożliwia utworzenie stosu o dowolnej pojemności zdefiniowanej przez użytkownika. Sprawdza, czy podana pojemność jest większa od 0. Jeśli nie, zgłasza wyjątek.

Destruktor:

- **~Stack()**
  Zwalnia pamięć zajmowaną przez dynamicznie alokowaną tablicę data.

Metody:

1. **void push(int value)**
   Dodaje element na wierzch stosu. Sprawdza, czy stos nie jest pełny; w przypadku przepełnienia zgłasza wyjątek std::overflow_error.
2. **void pop()**
   Usuwa element z wierzchołka stosu. Sprawdza, czy stos nie jest pusty; w przeciwnym razie zgłasza wyjątek std::underflow_error.
3. **int size() const**
   Zwraca liczbę elementów znajdujących się obecnie na stosie.

4. **bool isEmpty() const**
   Sprawdza, czy stos jest pusty. Zwraca wartość true, jeśli stos jest pusty, w przeciwnym razie false.
5. **void clear()**
   Opróżnia stos, ustawiając wskaźnik wierzchołka na -1.
6. **int top() const**
   Zwraca wartość elementu znajdującego się na wierzchu stosu. Jeśli stos jest pusty, zgłasza wyjątek std::underflow_error.
7. **void display() const**
   Wyświetla zawartość stosu w konsoli w postaci listy elementów od spodu do wierzchołka. Ułatwia wizualizację działania stosu.

Każda metoda została zaprojektowana z myślą o obsłudze błędów, aby zapewnić niezawodne działanie biblioteki w różnych scenariuszach.