Projektowanie algorytmów i metody sztucznej inteligencji

Przemysław Erbert 258964

wt 15:15

1 Link do repozytorium

https://github.com/Przemoerb/Pamsi1

2 Opis poszczególnych klas

W programie zdefiniowane sa dwie klasy: Element i Stack.

Klasa Element reprezentuje pojedynczy element stosu jako dynamiczna strukture danych, która zawiera wartość typu int oraz wskaźnik na nastepny element (typu Element*). Struktura ta jest używana tylko przez klase Stack.

Klasa Stack reprezentuje abstrakcyjny stos, który może być zaimplementowany jako stos na liście lub stos na tablicy. Klasa ta posiada dwie prywatne zmienne: int* arrayStack oraz Element* listStack. Pierwsza z nich reprezentuje tablice elementów stosu, druga reprezentuje liste elementów stosu. Zmienna arraySize przechowuje aktualna liczbe elementów na stosie.

3 Opis poszczególnych funkcji

Klasa Stack posiada publiczne funkcje, które umożliwiaja operacje na stosie. Funkcje pushElementToList, popElementFromList, clearList i printList umożliwiaja operacje na stosie jako liście. Natomiast funkcje pushElementToArray, popElementFromArray, clearArray i printArray umożliwiaja operacje na stosie jako tablicy. Funkcje pushElementToList i pushElementToArray dodaja element na stos, popElementFromList i popElementFromArray usuwaja ostatnio dodany element ze stosu, clearList i clearArray usuwaja wszystkie elementy ze stosu, natomiast printList i printArray wyświetlaja zawartość stosu na standardowym wyjściu.

- Stack::Stack() Konstruktor klasy Stack inicjuje wskaźniki listStack i arrayStack na wartość nullptr, a arraySize na 0.
- Stack:: Stack() Destruktor klasy Stack zwalnia pamieć zaalokowana na tablice arrayStack oraz elementy listy listStack.

- void Stack::pushElementToList(int value) Metoda pushElementToList dodaje nowy element do listy listStack. Tworzy nowy element, inicjuje jego wartość przekazanym argumentem value, a nastepnie przypisuje go jako nowa głowe listy.
- void Stack::popElementFromList() Metoda popElementFromList usuwa pierwszy element z listy listStack. Pobiera wskaźnik na głowe listy, przypisuje jej nastepnik jako nowa głowe, usuwa pierwotna głowe.
- void Stack::clearList() Metoda clearList usuwa wszystkie elementy z listy listStack. Wywołuje metode popElementFromList aż do momentu, gdy lista bedzie pusta.
- void Stack::printList() Metoda printList wypisuje wszystkie elementy listy listStack na standardowe wyjście. Przechodzi przez liste za pomoca wskaźnika tmp, wyświetla wartość elementu i przesuwa wskaźnik na nastepny element.
- void Stack::pushElementToArray(int value) Metoda pushElementToArray dodaje nowy element na koniec tablicy arrayStack. Zwieksza rozmiar tablicy o 1, alokuje nowa tablice o rozmiarze zwiekszonym o 1 i kopiuje elementy ze starej tablicy do nowej. Nastepnie usuwa stara tablice i przypisuje nowa jako arrayStack.
- void Stack::popElementFromArray() Metoda popElementFromArray usuwa ostatni element z tablicy arrayStack. Zmniejsza rozmiar tablicy o 1, alokuje nowa tablice o rozmiarze zmniejszonym o 1 i kopiuje elementy ze starej tablicy do nowej (bez ostatniego elementu). Nastepnie usuwa stara tablice i przypisuje nowa jako arrayStack.
- void Stack::clearArray() Metoda clearArray usuwa wszystkie elementy z tablicy arrayStack. Ustawia rozmiar tablicy na 0.
- void Stack::printArray() Metoda printArray wypisuje wszystkie elementy tablicy arrayStack na standardowe wyjście. Przechodzi przez tablice od ostatniego elementu do pierwszego i wyświetla wartość każdego elementu.
- void checkTimeArray(Stack s, int numberOfOccurrences) Metoda checkTimeArray mierzy czas potrzebny na wykonanie określonej liczby operacji na stosie implementowanym za pomoca tablicy. Dodaje numberOfOccurrences losowych element

4 Wynik działania programu

Stos jako lista:
czas dla 10 danych to 1 milisekund
czas dla 100 danych to 0 milisekund
czas dla 1000 danych to 1 milisekund

czas dla 10000 danych to 2 milisekund
czas dla 100000 danych to 8 milisekund
czas dla 1000000 danych to 79 milisekund
Stos jako tablica:
czas dla 10 danych to 1 milisekund
czas dla 100 danych to 1 milisekund
czas dla 1000 danych to 3 milisekund
czas dla 10000 danych to 217 milisekund
czas dla 100000 danych to 18074 milisekund