Лабораторна робота №3. Програмна реалізація АТД список, стек

Мета: розробити алгоритм і програму реалізації абстрактного типу даних список та стек.

Теоретичні відомості

Абстрактний тип даних (АТД) – це математична модель деякого об'єкта з сукупністю операторів, визначених в рамках цієї моделі.

Списки є надзвичайно гнучкою структурою, тому що їх легко зробити більшими або меншими, і їхні елементи доступні для вставки або видалення в будь-якій позиції списку. Списки також можна поєднувати або розбивати на менші списки. Списки регулярно використовуються в додатках, наприклад у програмах інформаційного пошуку, трансляторах мов програмування або при моделюванні різних процесів.

Лінійний список представляє собою послідовність елементів певного типу, що у загальному випадку будемо позначати як elementtype (тип елемента). Ми будемо далі представляти список у вигляді послідовності елементів, розділених комами: $a_1,a_2,...,a_n$, $n > 0$ де всі a_i, мають тип elementtype. Кількість елементів n будемо називати довжиною списку. Якщо n >1, то a_1 називається першим елементом, а a_n — останнім елементом списку. У випадку n = 0 маємо порожній список, що не містить елементів. Важлива властивість списку полягає в тім, що його елементи можна лінійно впорядкувати відповідно до їх позиції в списку.

Тепер перейдемо до безпосереднього визначення множини операторів списку.

- `INSERT(x,p,L)`. Вставити елемент `x` у позицію `p` у списку `L`.
- `LOCATE(x,L)`. Повертає позицію елемента `x` у списку `L`.
- `RETRIEVE(p,L)`. Повертає елемент, що стоїть в позиції `p` списку `L`.
- `DELETE(p,L)`. Цей оператор видаляє елемент у позиції `p` списку `L`.
- `NEXT(p,L)`, `PREVIOUS(p,L)`. Ці функції повертають відповідно наступну й попередню позиції від позиції `p` у списку `L`.
- `MAKENULL(L)`. Ця функція робить список `L` порожнім.
- `FIRST(L)`. Ця функція повертає першу позицію в списку `L`.
- `PRINTLIST(L)`. Друкує елементи списку `L` у порядку їх розташування.

При реалізації списків за допомогою масивів елементи списку розташовуються в суміжних комірках масиву. Таке розташування дозволяє легко переглядати вміст списку й вставляти нові елементи в його кінець. Але

вставка нового елемента в середину списку вимагає переміщення всіх наступних елементів на одну позицію до кінця масиву, щоб звільнити місце для нового елемента. Видалення елемента також вимагає переміщення елементів, щоб закрити комірку, що звільняється.

У багатьох додатках виникає необхідність організувати ефективне переміщення за списком як у прямому, так і у зворотному напрямках. Тому, важливою задачею є ефективна реалізація операцій над списками та збереження даних списку.

Стек – частинний випадок лінійного списку, у якому додавання та видалення елементів відбувається з одного кінця списку (**верхівка стеку**). Для позначення способу оперування з даними стека використовується абревіатура **LIFO** (англ. Last In First Out – останнім прийшов – першим вийшов). Інтуїтивною моделлю стека може бути стос тарілок – додати та зняти тарілку можна тільки з верхівки. Абстрактні дані типу стек зазвичай використовують наступний набір операторів.

- `MAKENULL(S)`. Робить стек порожнім.
- `TOP(S)`. Повертає елемент з верхівки стека. Якщо верхівка стека має позицію 1, тоді цей оператор над стеком можна переписати через оператори над списком таким чином: `RETRIEVE(FIRST(S),S)`.
- `POP(S)`. Повертає та видаляє елемент з верхівки стеку (виштовхує елемент зі стеку). Може бути реалізований через оператори списку як послідовність `RETRIEVE(FIRST(S),S)` та `DELETE(FIRST(S),S)`.
- `PUSH(x,S)`. Додає елемент `x` у верхівку стеку `S`. Елемент, який до цього був на верхівці стає наступним за верхівкою і т.д. Може бути реалізований через оператори списку як послідовність `INSERT(x,FIRST(S),S)`.
- `EMPTY(S)`. Ця функція повертає значення true, якщо стек `S` порожній та false в протилежному випадку.

Завдання до лабораторної роботи

I. Реалізувати АТД лінійний список. Провести обчислювальні експерименти з основними операціями над списком.

Для виконання поставленої мети, необхідно вирішити наступні задачі:

1. Реалізувати структуру даних для зберігання елементів списку.
2. Реалізувати основні операції роботи зі списком.

3. Провести математичний аналіз операцій роботи зі списком. При необхідності провести емпіричний аналіз для підтвердження результатів математичного аналізу.
4. Реалізувати інтерфейс користувача. Функціональні можливості інтерфейсу: виводити запрошення на уведення кількості елементів списку *n;* виводити послідовність операцій роботи зі списком з можливістю вибору користувачем необхідної операції.
5. Зробити висновок про ефективність реалізації списку за допомогою статичних структур даних.

II. Визначити, чи є введена строка паліндромом (наприклад, «404» - паліндром). При розв'язанні задачі використовувати АТД стек.

Контрольні запитання

1. Що таке абстрактний тип даних? Які АТД ви знаєте?
2. Які операції характерні для АТД список.
3. Привести приклади використання списків у програмуванні при вирішенні практичних задач.
4. Які існують підходи до реалізації списків? Оцініть їхню ефективність.
5. Обчисліть значення наведених нижче сум:

А) $\sum_{i=3}^{n+1} 1$, Б) $\sum_{i=3}^{n+1} i$, В) $\sum_{i=0}^{n-1} i(i+1)$, Г) $\sum_{j=1}^{n} 3^{j+1}$, Д) $\sum_{i=1}^{n}\sum_{j=1}^{n} ij$.

6. Визначите порядки росту наведених нижче сум. Використайте позначення $\Theta(g(n))$ з найпростішою функцією $g(n)$.

А) $\sum_{i=0}^{n-1} (i^2+1)^2$, Б) $\sum_{i=2}^{n-1} \lg i^2$, В) $\sum_{i=1}^{n} (i+1)2^{i-1}$, Г) $\sum_{i=0}^{n-1}\sum_{j=0}^{i-1} (i+j)$.

7. Знайдіть розв'язки наступних рекурентних рівнянь:

А) $x(n) = x(n-1) + 5$, для *n>1, x(1) =0.*

Б) $x(n) = 3x(n-1)$, для *n>1, x(1) =4.*

В) $x(n) = x(n-1) + n$, для *n>0, x(0) =0.*