

Class 49: Stacks

Held: Monday, 27 November 2006

Summary: Today we consider abstract data types in general and a particular abstract data type, the *stack*.

Related Pages:

- EBoard.
- Lab: Stacks.
- Reading: Stacks.

Notes:

- Tons of cool talks this week: Today, 1:15 (2417): Eye Tracking; Today, 4:15 (CCC): Assistive Technologies for Art; Thursday, 11:00 (2424): Interactive Character Animation; Thursday, 9:00 p.m. (2424): Rockstar Games Presents Table Tennis; Friday, 4:15 p.m. (2424): Careers in the Gaming Industry.
- Happy purgatory week!
- Are there questions on the exam?

Overview:

- Abstract Data Types (ADTs).
- Stacks.
- Implementing Stacks in Scheme.
- Some Applications.
- Lab.

Abstract Data Types

- As you've noted, when working with data we find many different ways to organize the data.
 - If we want to access information quickly by number, we use a vector.
 - If we want to add and remove information, we use a list.
 - If we want to access information by name, we use an association list.
- Good programmers decide on the key operations they wish to perform and then find the best possible implementation of those operations.
- A natural set of related operations is often called an *abstract data type*.
- The corresponding implementation of those operations is often called a *data structure*.

Stacks

- Stacks are an abstract data type with a few simple operations.
- Think about a stack of books or papers on a desk.
- You can add something to the top of the stack. Computer scientists call this operation *push*.
- You can remove something from the top of the stack. Computer scientists call this operation *pop*.
- You can look at the top of the stack. Computer scientists call this operation *top* or *peek*.

Implementing Stacks

- It is fairly easy to implement stacks in Scheme.
- We represent the stack as a list, with the newest item at the front.
- To push something, we cons it onto the front.
- To pop something, we cdr the list.
- How do we change a list in place? We don't, really. Instead, we store the list in a vector, and reset the contents of the vector.
- To make things even more fun, let's use object-oriented programming techniques.

Some Applications

- RPN calculator.
- Eliminating deep recursion.
- Matching tags in HTML.

Laboratory

- Attempt the lab.

Copyright © 2006 Samuel A. Rebelsky. This work is licensed under a Creative Commons Attribution-NonCommercial 2.5 License. To view a copy of this license, visit <http://creativecommons.org/licenses/by-nc/2.5/> or send a letter to Creative Commons, 543 Howard Street, 5th Floor, San Francisco, California, 94105, USA.