CSE2031 Software Tools - Structures and Unions

Summer 2010

Przemyslaw Pawluk

Department of Computer Science and Engineering
York University
Toronto

June 1, 2010

What have we done last time?

- Memory Allocation
- Structures

Table of contents

Plan

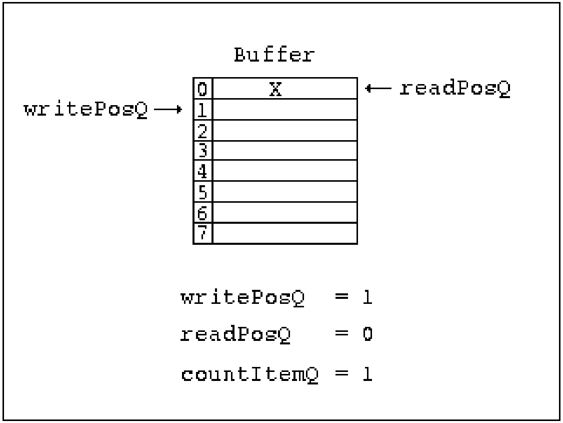
Linked List

Widely used structure i.e. to implement queues.

- Has head and tail
- head is a list element
- tail is a list
- each element points to the next one
- last element points to NULL

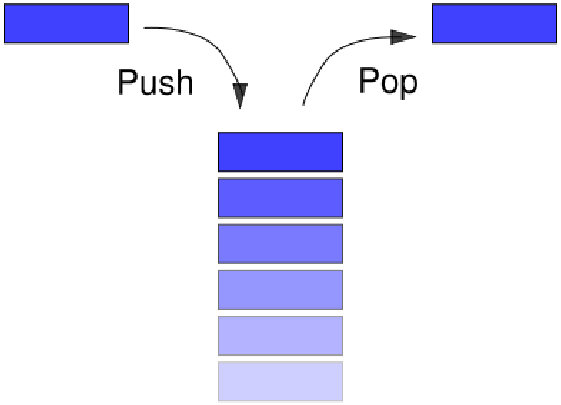
FIFO queue

- elements are added at the end
- elements are taken from the beginning



LIFO queue - Stack

- elements are added at the top
- elements are taken from the top

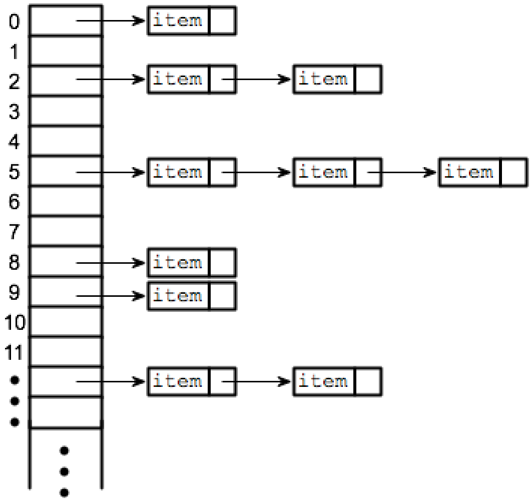


Hash-table – Table lookup

Hash-table in this case contains pointers to linked lists

- Flexible structure used to store multiple elements (i.e. text)
- Improves search
- Two methods are provided:
  - `install(s,t)` – adds element `s` and replacement text `t`
  - `lookup(s)` – looks for `s` in our hash-table

Hash Table


0
1
2
3
4
5
6
7
8
9
10
11
item
item
item
item
item
item
item
item
item
item

How to calculate hash?

- Hash is small number (between 0 and `HASH_SIZE`)
- Hash is calculated by hash function
- Hash is calculated based on value

Example

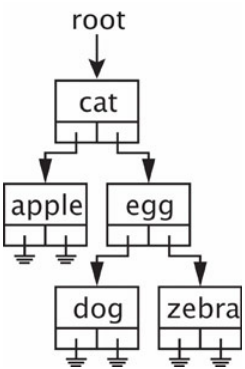
```
unsigned hash(char *s){
    unsigned hashval;

    for(hashval=0; *s!='\0'; s++)
        hashval=*s + 31 * hashval;

    return hashval % HASHSIZE;
}
```

Binary Tree - dictionary

- Tree has exactly one root element
- Each element has at most two children
- Each element stores a word and its translation
- Left child is less-or-equal than parent
- Right child is grater than parent



Good to know about structures

- structure may contain virtually instances of any type but ...
- **structure cannot contain an instance of itself**,
- structure can contain a pointer to itself,
- the size of structure is **not** necessarily equal to the sum of sizes of members (depends on implementation)

Plan

Union

- Variable that may hold at different times objects of different types and sizes
- Compiler keeps track of size and alignment requirements
- It's programmer's responsibility to keep track of which type is currently stored in a union
- type retrieved must be the one most recently stored
- implementation-dependent results if something is stored as one type and retrieved as another
- access to the union members is syntactically the same as to structure members `union.member` or `union_ptr->member`
- union is large enough to store "widest" member
- all members are stored in the same are in memory

Union

Initialization

Union may only be initialized with a value of the type of its first member!

Union init. - example

Correct

```
union {
    int x;
    float y;
    char *sptr;
} u = 1;
```

Incorrect

```
char s[] = "test";
union {
    int x;
    float y;
    char *sptr;
} u = &s;
```

Union init. - example

Correct

```
union {
    int x;
    float y;
    char *sptr;
} u = 1;
```

Incorrect

```
char s[] = "test";
union {
    int x;
    float y;
    char *sptr;
} u = &s;
```

Plan

Bit-fields

- Used when storage space is at premium
- Allows to pack several objects into a single machine word
- Can be used to implement flags or masks

Masks using define

```
#define KEYWORD  01  /*0000 0001*/
#define EXTERNAL 02  /*0000 0010*/
#define STATIC   04  /*0000 0100*/
```

Masks using bit-fields

```
struct{/*one bit per flag*/
    unsigned int is_keyword : 1;
    unsigned int is_extern  : 1;
    unsigned int is_static  : 1;
} flags;
```

Plan

C-basics

- Program structure
- variables' types
- mixed type arithmetic
- cast
- precedence of operators
- conditional expressions
- pre- vs. post-
- numbers in C

I/O

- Char by char I/O
  - getchar()
  - putchar()
- Formated I/O
  - printf()
  - scanf()
  - different formatters %s, %d, %f, %c ...

Preprocessor

- #declare
- #include
- #if, #elif, #else and #endif
- defined(name), #ifdef and #ifndef

Scope

- Declaration vs. definition
- scope
- external vs. internal
- static and extern

Testing

- Random tests
- Black-box tests
- Glass-box tests
- Regression tests
- Boundary conditions testing
- Pre- and Post-condition testing
- Assertions

Arrays

- Definition
- Initialization
- Access
- Size

Pointers

- Definition
- Access to a pointee
- Getting the address of a variable
- Arithmetic
- Pointers and Arrays
- void*

Different definitions

- to simple types
- to arrays, structures and unions
- to functions

Memory allocation

- malloc
- calloc
- realloc
- free

Structures and Unions

- Structures
  - Definition
  - Linked Lists (FIFO, stack)
  - Trees
  - Hash tables
  - bit-fields
- Unions
  - Definition
  - Properties
- initialization
- namespace
- deep vs. shallow copy

Lab-test

- 3 programming tasks
- standard I/O
- arrays, structures and unions
- memory allocation