CS 180 - Queues

Announcements

- HW due Monday by start of class
- Next assignment will be out Friday (program on stacks)
Queue

list - stores in FIFO

(like a line)

(stacks were LIFO)
Alright - let's think about the setup:

```cpp
template <typename Object>
class Queue {
public:
    int size() const;
    bool isEmpty() const;
    const Object& front() const;
    void enqueue(Object obj);
    Object dequeue();
};
```
How to implement?

use an array

Private data:

```
Object* obj;
int f;
int size;
int capacity;
```

in constructor

\( \text{obj} = \text{new Object \[\text{copy}\]} \)
Wrapping Around:

Q
0 1 2 3 4 ...

Capacity
f + size

Q
Two options:

- A lot of if statements

- Modular arithmetic: remainders

\[
\begin{align*}
1 \mod 3 &= 1 \\
4 \mod 3 &= 1 \\
5 \mod 3 &= 2 \\
11 \mod 3 &= 2
\end{align*}
\]
**Pseudo code**

is Empty():

    return (size == 0);

size():

    return size;
enqueue (element):

if \((f + \text{size}) \mod \text{capacity} = f\)

then throw error 0

else

\(Q[(f + \text{size}) \mod \text{capacity}] = \text{element}\)

\(\text{size}++\)

\(f = f + \text{size} \mod \text{cap}\)

\(\text{size}=16\)
Object deq

eu e ( ) {} 

if (is Empty())
    throw Error;
else
    int old f = f;

    f = (f + 1) % capacity;
    size --;
    return Q[old f];
Actual code
(on webpage or in text)
Housekeeping Functions:

- Destructor
- Operator =
- Copy constructor

private:

void CopyData(const Array Queue & other) {
    // assuming this and other have some capacity
    f = other.f;
    int walk = other.f;
    for (int i = 0; i < size; i++) {
        Q[i] = other.Q[walk];
        walk = (walk + 1) % capacity;
    }
}
Array Queue & operator=(const Array Queue & other) {
    if (this != & other) {
        size = other.size;
        capacity = other.capacity;
        delete[] Q;
        Q = new Object[capacity];
        copy Data (other);
    }
    return *this;
}

stack1 = stack2;
Array Queue (Q)

3

Array Queue (Q)

Array Queue (Q)

(q)

Array Queue (const Array & other)

Stack 2 (stack 2)

Array Queue (Q)

Array Queue (const Array & other)

Stack 2 (stack 2)