* Ch 8 vectors contine
  + Lab 4 functions
    - Template Class
    - Capacity vs size
    - Copy constructor – explicit parameter comes in and the implicit becomes exact copy for that. Delete the implicit buffer then create the new implicit from the explicit
    - Vector – int size, int cap, T \*buffer.
    - Begin() returns the address of first location.
    - First method
      * Template <class T>
      * Typename Vector<T>::iterator Vector <T>:: begin()
        + Return buffer;
    - Second method
      * Template <class T>
      * Pop back just decrement the size
      * Push back put value in the back then increase size.
      * If size == cap, create a new T[cap + 1]. Then copy vector over to the new buffer. Then add new item to the end. Maybe increase the size to +5 or more just to remove the compile time.
    - Third
      * Template<class T>
      * Void Vector<T>::reserve(unsigned int capacity)
        + If (capacity <=my\_capacity)

Return;

* + - * + Else

My\_capacity = capacity;

T \*new\_buffer = new T[capacity];

For(int size = 0; size < my\_size < size++)

New\_buffer[size] = buffer[size];

Delete[] buffer;

Buffer = new\_buffer;

* + - Note heap --> dynamic memory. Are the same terms.
  + Chapter 2 – War Program
    - Card game
    - 2 player game
    - Deck of regular cards.
    - Each player gets 3 cards initially. Players draw one of their cards Who ever has the highest card wins(2 points). A tie results in 1 point to each player. Players replace their drawn card from the deck. Game ends when player hands are empty.
    - Ace has lowest rank... king has highest rank.
    - Steps in a object oriented programming/design
      * Identify the classes and the methods.
        + Nouns are classes
        + Verbs are methods.
    - Bottom up design.
      * Reuse encapuslation
      * Class Card
        + Int rank;
        + Int suit; --> Suits suit;

Enum (enumeration) Suits{diamonds, clubs, hearts, spade};

* + - * + Public:

Card();

Card(int s, int r);

* + - * Class Deck
        + Card cards[52];
        + Int topCard;
        + Public:

Deck();

Card draw();

Void shuffle();

Bool isEmpty(); --> return topCard==0;

* + - * UML – unified modeling language
        + Deck ( - is private, + is public

Variables

-TopCard: integer

Methods

+ Deck()

+ draw():Card

* + - * + Field Diamond
        + 1.52 - cards
        + Card

Variables

-rank:Integer

Methods

+Card();

+Card(Suits, integer)

* + - * + Field diamond
        + 1.1 - suits
        + Suits

Variable

<<enumeration

Suits

Method

+diamonds

+spades

+clubs

+Hearts