Lab 5

Announcement

- No more collaborations on code implementation
  - You can still share test cases with your partner(s)

Project: String

- Private variables
  - char *characters;
  - size_t allocated;
  - size_t current_length;

Project: String

- FIRST, implement reserve(size_t n)
  - Postcondition: All functions will now work efficiently (without allocating new memory) until n characters are in the string.
  - You don’t need to worry about reducing size.
  - Situation 1: n==allocated (optional)
  - Situation 2: n<current_length+1
  - Situation 3: Otherwise
  - 1. Allocate new memory; 2. Copy current content; 3. Delete old memory; 4.Pointing to new memory

Project: String

- Constructor & Destructor
  - string(const char str[ ] = ""); //Post Condition: The string contains the sequence of chars from str
    1. Get the string length; 2. Reserve memory; 3. Copy content. (Don’t forget ‘/0’)
  - string(char c); //Postcondition: The string contains c and then the null character.
  - string(const string& source); //Postcondition: The string becomes a copy of the source string.
  - ~string( ); //Delete memory & Reset variables

Project: String

- Constant Member Functions
  - size_t length( ) const { return current_length; }
  - char operator [ ](size_t position) const;
  - int search(char c) const; //Postcondition: The location of the FIRST occurrence of the character c within this string is returned. If the string does not contain c, -1 is returned.
  - int search(const string& substring) const; //Postcondition: Returns the index of the start of the first instance of the given substring inside of this string. If the substring is not found, this function returns -1.
    Library Function: strstr()
  - unsigned int count(char c) const; //The count of the occurrence of the character c within this string is returned.

Project: String

- Modification Member Functions
  - void operator +=(const string& addend); //reserve() + strncat()
  - void operator +=(const char addend[ ]); //reserve() + strncat()
  - void operator +=(char addend); //reserve()
  - void reserve(size_t n);
  - string& operator =(const string& source); //reserve() + strncpy()
  - void insert(const string& source, unsigned int position); //reserve() + strncpy() + strncat()
  - void dlt(unsigned int position, unsigned int num);
  - void replace(char c, unsigned int position);
  - void replace(const string& source, unsigned int position);

Project: String

- Friend Non Member Functions
  - friend std::ostream& operator <<(std::ostream& outs, const string& source); //The sequence of characters in source has been written to outs. The return value is the ostream outs.

  //The six comparison operators (==, !=, >=, <=, >, and <) are implemented for the string class, forming a total order semantics, using the usual lexicographic order on strings.
  - friend bool operator ==(const string& s1, const string& s2);
  - friend bool operator !=(const string& s1, const string& s2);
  - friend bool operator > (const string& s1, const string& s2);
  - friend bool operator < (const string& s1, const string& s2);
  - friend bool operator >=(const string& s1, const string& s2);
  - friend bool operator <=(const string& s1, const string& s2);

Project: String

- Non Member Functions
  - string operator +(const string& s1, const string& s2); //use operator+=
  - string operator +(const string& s1, const char addend[ ]); //use operator+=
  - std::istream& operator >> (std::istream& ins, string& target); //Postcondition: A string has been read from the istream ins, and the istream ins is then returned by the function.

Don’t Forget

- Demo code to me
  - Today or next week
  - Must compile and run on linux servers
- Comment code
- File with description of lab is on Camino
  - Submission guidelines