

## Day 26 (Wed 03/30)

- exercise 25 review
- day 26 recap questions
- exercise 26

## Announcements/reminders

- \* - Please submit midterm project survey (on Gradescope) no later than Friday 4/8
- \* - HW5: due Wednesday 4/6 by 11pm

## Exercise 25

abbrev function loop

```
for ( size_t i = 0; i < word.size(); i++ ) {  
    bool cur_is_vowel = is_vowel(word[i]);  
  
    if ( cur_is_vowel ) {  
        if ( !last_was_vowel ) {  
            result += " ";  
        }  
    } else {  
        result += word[i];  
    }  
  
    last_was_vowel = cur_is_vowel;  
}
```

## Exercise 25

main function, opening input and output files

```
{  
    ifstream in(argv[1]);  
    if ( !in.is_open() ) {  
        cerr << "Couldn't open input file " << argv[1] << endl;  
        return 1;  
    }  
  
    ofstream out(argv[2]);  
    if ( !out.is_open() ) {  
        cerr << "Couldn't open output file " << argv[2] << endl;  
        return 1;  
    }  
}
```

fopen (C)

vs

ifstream/  
ofstream (C++)

```
void read_stuff(  
    ifstream &in)  
{  
    ;  
}
```

...  
\*  
}

## Exercise 25

using std::getline;

main function: main loop

```
string line;
while (getline(in, line)) {
    stringstream line_in(line);
    string word;
    while (line_in >> word) {
        out << abbreviate(word) << " ";
    }
    out << endl;
}
```

## Exercise 25

classify program

body of loop, variables

```
double fpval;
```

```
int ival;
```

```
string extra;
```

```
bool is_ival = false, is_fpval = false;
```

## Exercise 25

123.45

classify program, body of loop

check whether token is an integer value:

```
stringstream as_i(token);
```

```
if (as_i >> ival) {
```

```
    if (!(as_i >> extra)) {
```

```
        sum_i += ival;
```

```
        is_ival = true;
```

```
    }
```

```
}
```

## Exercise 25

classify program, body of loop

determine whether token is a floating point value

```
if (!is_ival) {  
    stringstream as_fp(token);  
    if (as_fp >> fpval) {  
        sum_fp += fpval;  
        is_fpval = true;  
    }  
}
```

## Exercise 25

classify program: handle tokens that are neither integer nor floating-point

```
if (!is_ival && !is_fpval) {  
    ntok++;  
    ntok_c += token.size();  
}
```



## Exercise 25

letter\_freq program: initialize vector of Buckets

```
vector<Bucket> buckets;  
for (int i = 0; i < 26; i++) {  
    Bucket b;  
    b.letter = 'a' + i;  
    b.count = 0;  
    buckets.push_back(b);  
}
```

```
struct Bucket {  
    char letter;  
    unsigned count;  
};
```

better: constructor

## Exercise 25

letter\_freq program: open input file, read characters, classify them

```
ifstream in(argv[1]);  
if (!in.is_open()) {  
    cerr << "Couldn't open input file " << argv[1] << endl;  
    return 1;  
}
```

```
char c;  
while (in.get(c)) {  
    c = tolower(c);  
    if (isalpha(c)) {  
        buckets[c - 'a'].count++;  
    }  
}
```

`ifstream::get(char &c)` {  
 ;  
 c = a\_char;  
}

## Exercise 25

letter\_freq program

Bucket comparison function

```
bool compare_buckets(const Bucket &left, const Bucket &right) {  
    if (left.count > right.count) { return true; }  
    if (left.count < right.count) { return false; }  
    return left.letter < right.letter;  
}
```

*tie-breaker*

Sorting the vector of Buckets:

```
sort(buckets.begin(), buckets.end(), compare_buckets);
```

*std::sort*

*comparator*

## Exercise 25

letter\_freq program: printing letter frequencies

```
for (vector<Bucket>::const_iterator i = buckets.cbegin();  
    i != buckets.cend();  
    i++) {  
    if (i->count > 0) {  
        cout << i->letter << ": " << i->count << endl;  
    }  
}
```

## Day 26 recap questions

1. What is a C++ reference?
2. When should you use C++ references?
3. What is the difference between a pointer and a reference?
4. How do you dynamically allocate memory in C++?
5. How do you free memory in C++?

# 1. What is a C++ reference?

A reference is an alias (alternate name) for a variable.

In its lifetime, a reference can ONLY refer to one variable.

Most common use: true reference parameters. E.g.:

```
void swap(int &a, int &b) { ...a and b are aliases for argument variables... }
```

Another important use: const reference parameters. Very useful for passing a large object or collection to a function, since it avoids copying. E.g.:

X void myfunc1(vector<int> a\_vec); // all elements from argument vector must  
by-value // be copied into a\_vec, could be very slow

✓ void myfunc2(const vector<int> &a\_vec); // a\_vec is an alias for the argument  
// vector: no copying required

```
int x = 2, y = 3;  
swap(x, y);
```

## 2. When should you use C++ references?

Allowing a function to have an alias to an argument variable, so it can modify the argument variable.

Accepting a const reference to an object where copying would be slow.

Occasionally: capture a reference to a collection element so you can modify it.

E.g.,

```
vector<int> myvec;
```

...

```
int &element = myvec[i];
```

```
element *= 2; // this modifies myvec[i]
```

### 3. What is the difference between a pointer and a reference?

#### Reference:

Does not require explicit address-of (&) to create, or explicit dereference (\*) to access the variable the reference refers to.

Cannot be reassigned. (It can only refer to one variable during its lifetime.)

#### Pointer:

Requires explicit address-of (&) to create, and explicit dereference (\*) to access the variable the pointer points to.

Can be reassigned. An assignment to a pointer variable changes what the pointer variable points to.



#### 4. How do you dynamically allocate memory in C++?

new or new[]

Dynamically create one variable:

```
int *p = new int;  
*p = 42;
```

Dynamically create an array:

```
int *p = new int[10];  
for (int i = 0; i < 10; i++) { p[i] = i; }
```

## 5. How do you free memory in C++?

delete or delete[]

Example:

```
int *p = new int;  
*p = 42;  
delete p;
```

Example:

```
int *p = new int[10];  
for (int i = 0; i < 10; i++) { p[i] = i; }  
delete[] p;
```

not recommended

```
int &p = *(new int);  
:  
:
```

```
delete &p;
```



