Note Title 05-11-2018

Last class:
Big O using recursive idea. HW:- solve the exercise problem in detail

COL100
NOV 5, 2018

Intuition:-

$x^n = x * x^{n-1}$
(↓ divide, ↓ self similar)

$O(n)$ $x^{18} = x * x^{17}$

① Intialy:- pow(x,n) :- x^n

Loop ✓ O(n)

```
int pow(int x, int n) {
    if(n == 0)
        return 1;
    int ans = 1;
    for(int i = 1; i <= n; i++){
        ans = ans * x;
    }
    return ans;
}
```

Recursive idea:

```
int pow(int x, int n){
    if(n == 0)            // Base case
        return 1;
    return x * pow(x, n-1);   // recursive sub-problem
}
```
(x: a unit; pow(x,n-1): recursive call)

$n = 2^k$

$x^0 = 1$ } Base case

$x^{18} = (x^2)^{18/2}$
$= (x^2)^9$

n even:-
$x^n = (x^2)^{n/2}$ } Recursive Case 1

$(y)^9 = y * (y)^8$

n: odd:-
$y^n = y * y^{n-1}$ } Recursive Case 2

```
int pow(int x, int n){
    if(n == 0)          // Base
        return 1;
    if(n % 2 == 0)      // Rec 1
        return pow(x * x, n/2);
    else                // Rec 2
        return x * pow(x, n-1);
}
```

$17 = 7 + 10^1 \times 1$

$781 = 10^0 \times 1 + 10^1 \times 8 + 10^2 \times 7$

$111_2 = 1 \times 2^0 + 1 \times 2^1 + 1 \times 2^2$

$= 7_{10}$

$101_2 = 1 \times 2^0 + 0 + 1 \times 2^2 = 5$

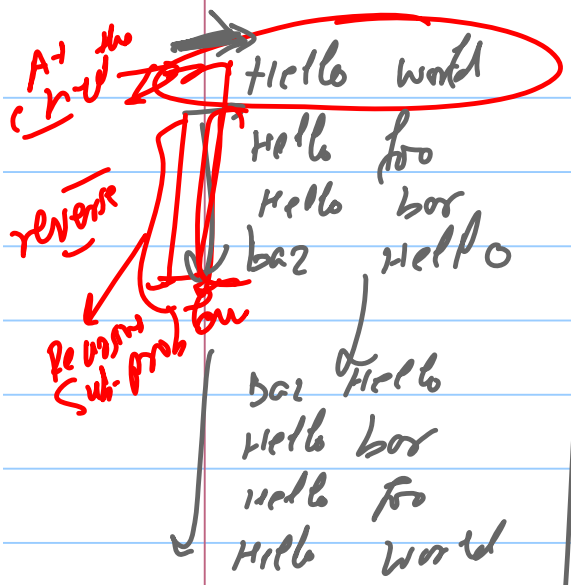
Binary representation of a number

```
int ConvertFromBinary(string str) {
    int len = str.length();
    if (len == 1)
        return stoi(str);
    int v0 = stoi(str.substr(len-1));
    string remaining = str.substr(0, len-1);
    return v0 + 2 * ConvertFromBinary(remaining);
}
```

$a_n a_{n-1} \cdots a_0$

$= 2^n a_n + 2^{n-1} a_{n-1} + \cdots + 2^0 a_0$

$= (2^{n-1} a_n + 2^{n-2} a_{n-1} + \cdots + 2^0 a_1) \times 2 + a_0$



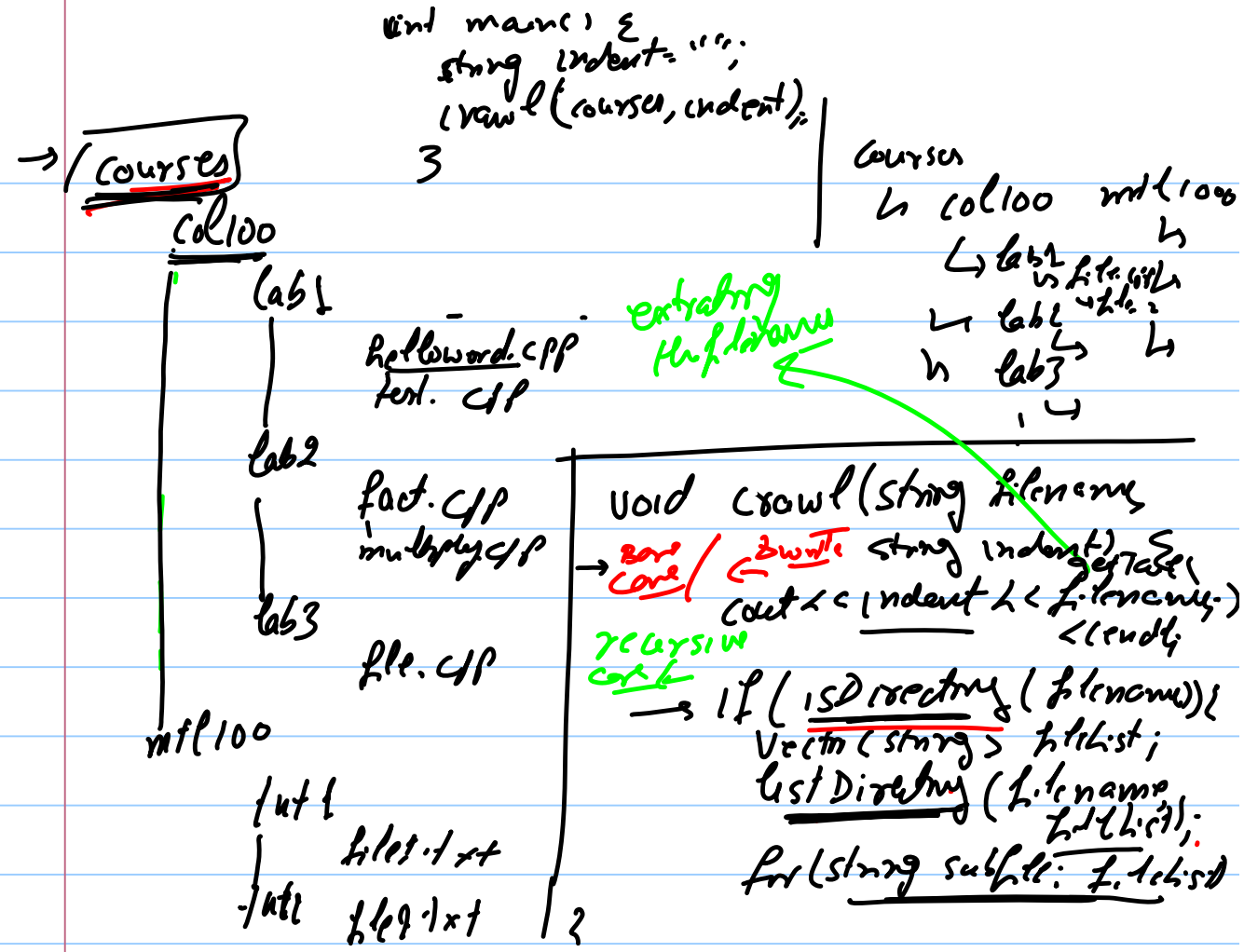

```
void reverseLines(ifstream &input){
    string line;
    if (getline(input, line)){
        reverseLines(input);
        cout << line << endl;
    }
}

int main(){
    ifstream input;
    input.open("file.txt");
    reverseLines(input);
}
```

Base Case

Recursive Case

Always


int main() {
string indent = "";
crawl(courses, indent);
}
courses
col100
mtl100
lab1
lab2
lab3
tut1
tut2
helloworld.cpp
test.cpp
fact.cpp
multiply.cpp
file.cpp
file1.txt
file2.txt
extracting the filename
void crawl(string filename, string indent) {
base case
cout << indent << filename << endl;
recursive case
if (isDirectory(filename)) {
vector<string> fileList;
listDirectory(filename, fileList);
for (string subfile : fileList)

```
for (String subfile: fileList) {
    crawl(filename + "/" + subfile,
          (indent + "    "));
}
```