16.3

7 best problems on Recursion

① Reverse a string using recursion

'binod'	print 'b'
'inod'	Print 'i'
'nod'	Print 'n'
'od'	print 'o'
'd'	print 'd'
''	Return

Code:

```
void reverse (string s) {
    if (s.length() == 0) { // base case
        return;
    }

    String ros = s.substr(1);

    reverse (ros);

    cout << s[0];
}
```

usage: reverse (string)

② JT Replace pi with 3.14 in string

"pippxxppiixipi"

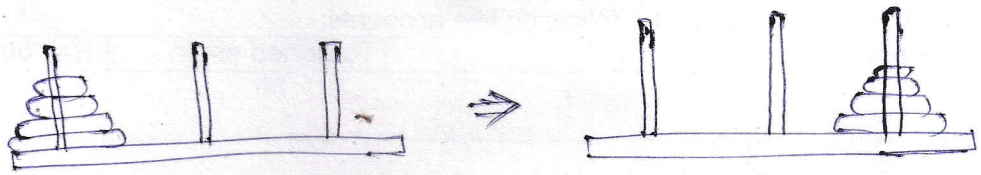
"3.14 ppxxp3.14 ixi 3.14"

Code:

```
void replacePi(string s){
    if(s.length() == 0){ // base case
        return;
    }
    if(s[0] == 'p' && s[1] == 'i'){
        cout << "3.14";
        replacePi(s.substr(2));
    }
    else {
        cout << s[0];
        replacePi(s.substr(1));
    }
}
```

usage: replacePi("pippppiiii pi");

③ Tower of Hanoi



Code:

```
void towerofHanoi(int n, char src, char dest, char helper){
    towerofHanoi(n-1, src, helper, dest);
    cout
    if (n==0){      // base case
        return;
    };

    towerofHanoi(n-1, src, helper, dest);
    cout << "Move from " << src << " to " << dest<<endl;
    towerofHanoi(n-1, helper, dest, src);
}
```

usage: `towerofHanoi(3, 'A', 'C', 'B');`

(4) Remove all duplicates ~~using~~ from the string

"aaaabbbeeecdddd"

"abecd"

Code:

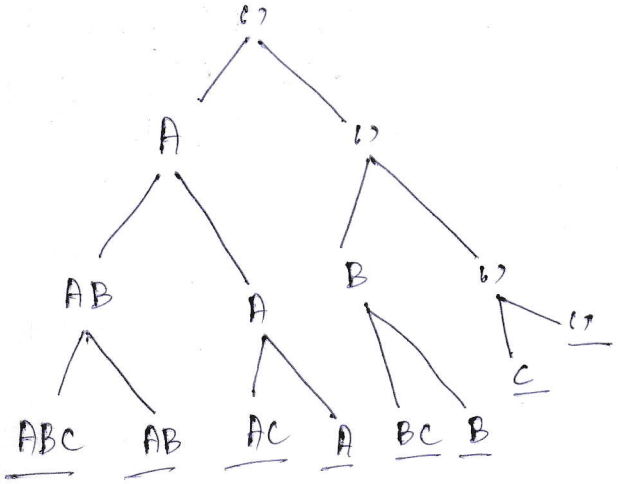
```
String removeDup (string s) {
    if (s.length() == 0){
    return "";
    }
    Char ch = s[0]
    String ans = removeDup (s.substr (1));
    if (ch == ans[0]){
        return ans;
    }
    return (ch + ans);
}
```

usage: cout << removeDup ("aaaabbbeeecdddd") << endl

⑥ Generate all substrings of a string

"ABC"

""
"A"
"B"
"AB"
"C"
"AC"
"BC"
"ABC"

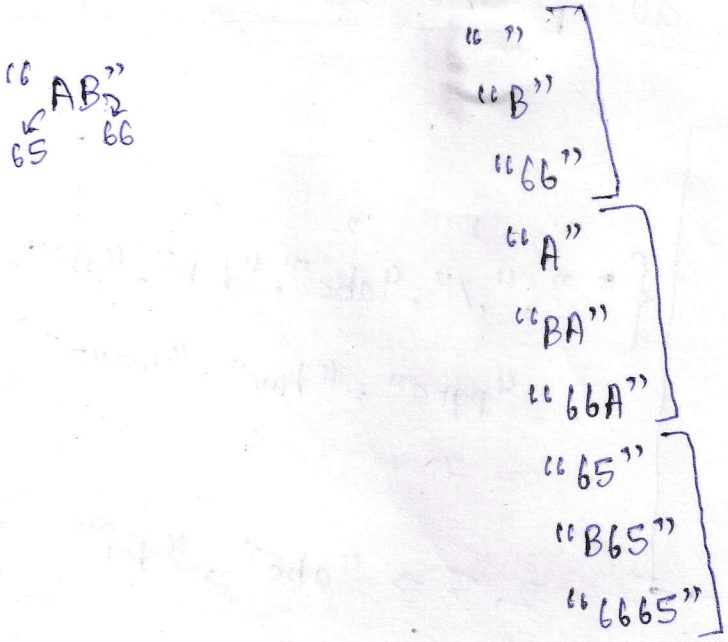


Code:

```
void subseq (string s, string ans) {
    if (s.length() == 0) {
        cout << ans << endl;
        return;
    }

    char ch = s[0];
    string ros = s.substr(1);

    subseq (ros, ans);
    subseq (ros, ans+ch);

}
```

Usage: subseq ("ABC", "");

Generate substrings with ASCII number



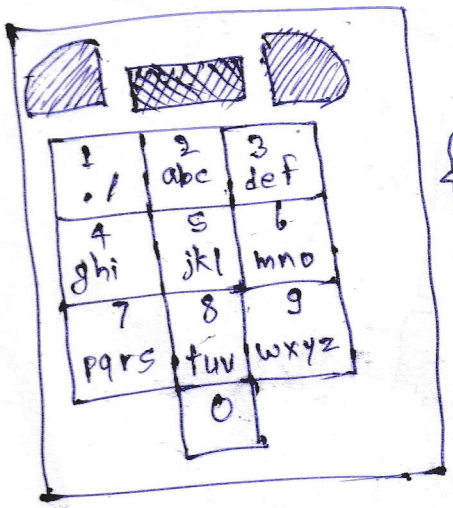
Code:

```
void subseq(string s, string ans) {
    if (s.length() == 0) {
        cout << ans << endl;
        return;
    }
    char ch = s[0];
    int code = ch;
    string ros = s.substr(1);
    subseq(ros, ans);
    subseq(ros, ans+ch);
    subseq(ros, ans + to_string(code));
}
```

usage: subseq("AB", "");

Keypad Problem

Print all possible words from phone digits



0 1 2 3 4 5 6

{"", "./", "abc", "def", "ghi", "jkl", "mno",

"pqrs", "tuv", "wxyz"}

7 8 9

2, 3 ⇒ "abc", "def" →

ad
ae
af
bd
be
bf
cd
ce
cf

Code:

```
string keypadArr[] = {"", "./", "abc", "def", "ghi"
                      "jkl", "mno", "pqrs", "tuv", "wxyz"};

void keypad(string s, string ans){
    if(s.length()==0){
        cout<<ans<<endl;
        return;
    }
    char ch = s[0];
    string code = keypadArr[ch-'0'];
    string ros = s.substr(1);
    for(int i=0; i<code.length(); i++){
        keypad(ros, ans+code[i]);
    }
}

usage: keypad("23", "");
```