*// HW4\_Q1*

#include <iostream>

using namespace std;

class node {

private:

node \*\_prev;

node \*\_next;

int \_data;

public:

node(){ *//建構子*

\_prev = NULL;

\_next = NULL;

\_data = 0;

};

node \*prev(){return \_prev;}; *//取出 \_prev*

node \*next(){return \_next;}; *//取出 \_next*

int data(){return \_data;}; *//取出 \_data*

void prev(node\* n){\_prev = n;}; *//寫入 \_prev*

void next(node\* n){\_next = n;}; *//寫入 \_next*

void data(int d) {\_data = d;}; *//寫入 \_data*

void show();

};

class list {

private:

node\* root\_node;

public:

list(int n){ *//建構子 依題目要求建構出陣列*

*/\*\**

*\* TODO*

*\* @var cur\_node 存放當前節點*

*\* @var new\_node 愈加入之新節點*

*\* 先令 cur\_node = root\_node*

*\* 隨後不斷往後走(for @var i=0)*

*\* 每次產生新節點放入new\_node，給值(2\*i+1)*

*\* new\_node 與 cur\_node 連接*

*\* 直到(i > n)*

*\* 或下個節點的直大於n(cur\_node->data() >= n)*

*\*/*

};

node\* root(){return root\_node;};

void show(); *//印出當前陣列，code在下部*

void add(int n);*//依題目要求加入節點，code在下部*

};

int main() {

*/\*\**

*\* TODO*

*\* 輸入list長度n*

*\* 判斷n*

*\* 產生list list(n)*

*\* 先另 cur\_node = root\_node*

*\* 隨後不斷往後走(while)*

*\* 直到下個節點為空(cur\_node->next()==NULL)*

*\* 或下個節點的直大於n(cur\_node->data() >= n)*

*\*/*

return 0;

}

void list::show(){

*/\*\**

*\* TODO*

*\* show all the node in the list*

*\*/*

}

void list::add(int n){

*/\*\**

*\* TODO*

*\* 插入數字n到list中*

*\* @var cur\_node 存放當前節點*

*\* @var new\_node 愈加入之新節點*

*\* 先令 cur\_node = root\_node*

*\* 隨後不斷往後走(while)*

*\* 直到下個節點為空(cur\_node->next()==NULL)*

*\* 或下個節點的直大於n(cur\_node->data() >= n)*

*\*/*

}

*// HW4\_Q3*

使用Backtracking實作

CODE解釋夠清楚了