//comsc-200

// main.cpp

// 12a

//

// Created by Jeff on 10/31/16.

// Copyright © 2016 Jeff zhang. All rights reserved.

//

#include <iostream>

#include <string>

#include <cstring>

using namespace std;

void Reverser(string a,int last\_index){

if(0 <= last\_index){

cout<<a[last\_index];

Reverser(a,last\_index-1);

}

else

return;

}

int main(int argc, const char \* argv[]) {

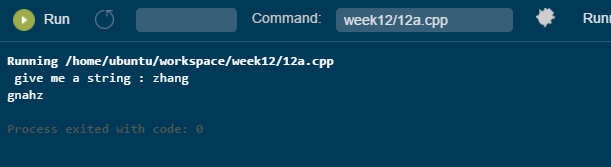
string a;

cout << " give me a string : " ;

getline(cin,a);

Reverser(a,a.length()-1);

}



//

// main.cpp

// 12b

//

// Created by Jeff on 10/31/16.

// Copyright © 2016 Jeff zhang. All rights reserved.

//

#include <iostream>

using namespace std;

int h = 0;

int rabbit ( int n){

string tabs(h,'\t');

if(n >h) h++;

if(n <=2){

cout << tabs << " Enter n = " << n << endl;

cout << tabs << "leave n = " << n << " instance <= 1" << endl;

return 1;

}

else{

cout <<tabs << " Enter n = " << n << endl;

// return rabbit(n-1)+rabbit(n - 2);

int x = rabbit(n-1)+rabbit(n - 2);

cout << "leave n = " << n << "instance = " << x << endl;

n--;

return x;

}

}

int main() {

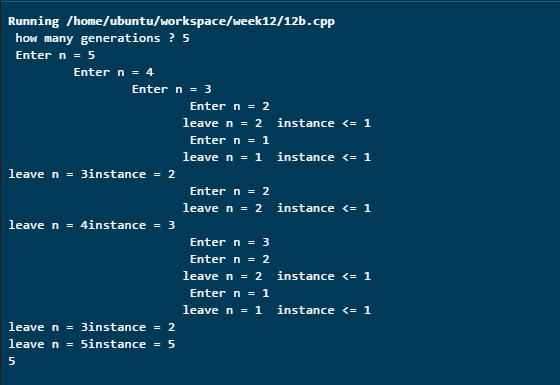
int gen;

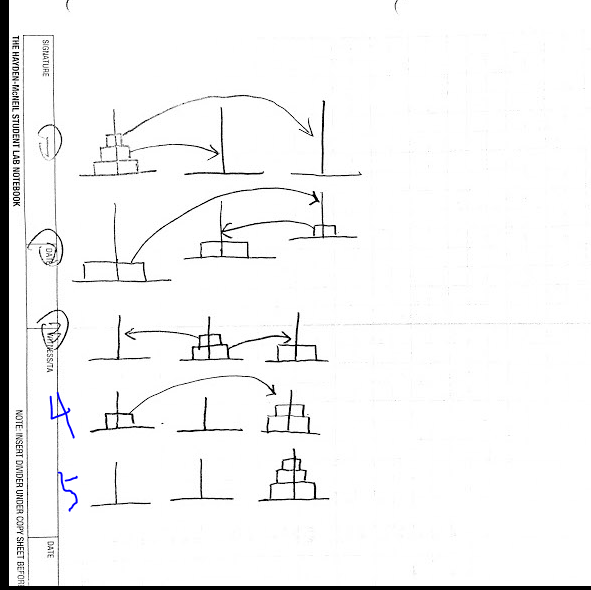
cout << " how many generations ? ";

cin >>gen;

cout << rabbit(gen) << endl;

}





// comsc 200

// main.cpp

// 12d

// boli zhang

// Created by Jeff on 11/2/16.

// Copyright © 2016 Jeff zhang. All rights reserved.

//

#include <iostream>

#include <deque>

#include <iterator>

using namespace std;

int main() {

deque<double>ddq;

cout << "add some numbers to the back, 0 to end: ";

istream\_iterator<double>dbl\_itr(cin);

while(true){

if(\*dbl\_itr == 0) break;

ddq.push\_back(\*dbl\_itr);

++dbl\_itr;

}

cout << " \n the deque: ";

for (auto item:ddq) cout << item << " ";

cout << "add some numbers to the front, 0 to end " ;

++dbl\_itr;

while(true){

if(\*dbl\_itr == 0) break;

ddq.push\_front(\*dbl\_itr);

++dbl\_itr;

}

cout << " \n the deque: ";

for (auto item:ddq)

cout << item << " ";

cout << "\nPop\_front; \n\tThe Deque: ";

ddq.pop\_front();

for (auto item:ddq)

cout << item << " ";

cout << "\nPop\_back; \n\tThe Deque: ";

ddq.pop\_back();

for (auto item:ddq)

cout << item << " ";

cout << "\ncopy constructor example\n";

deque<double> values(ddq);

cout << "\n\tthe duplicated deque: ";

for (auto item:ddq)

cout << item << " " ;

cout << endl;

ddq.pop\_front();

ddq.pop\_front();

cout << "\nddq[1] = 99.9\n";

cout << "\n\tthe Deque: ";

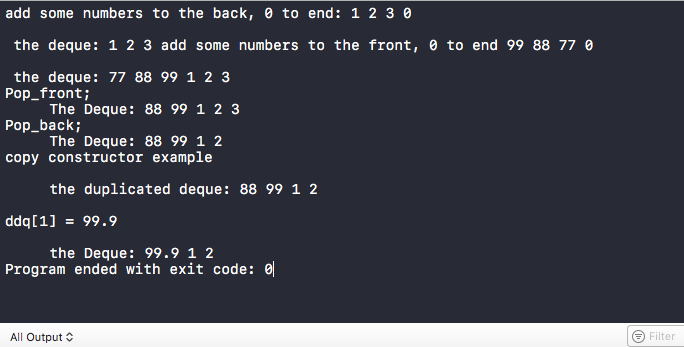
ddq.push\_front(99.9);

for (auto item:ddq)

cout << item << " ";

cout << endl;

}



//

// main.cpp

// 12e

//

// Created by Jeff on 11/2/16.

// Copyright © 2016 Jeff zhang. All rights reserved.

//

#include <iostream>

#include <array>

#include <vector>

#include <list>

#include <stack>

#include <queue>

#include <deque>

#include <map>

#include <set>

using namespace std;

int main(int argc, const char \* argv[]) {

//-----------------array----------------

cout << "\n1. int array \n";

int iArr[] = {1,2,3,4,5};

for (auto item:iArr) cout << item << " ";

cout << endl;

//------------------vector---------------

cout << "\n2. int vector \n";

vector<int>iVec = {1,2,3,4,5};

for (auto item:iVec) cout << item << " ";

cout << endl;

//-----------------string-----------------

cout << "\n3. string list \n";

vector<string>ilist= {"a","b","c"};

for (auto item:ilist) cout << item << " ";

cout << endl;

//----------------------stack----------------

cout << "\n4. string stack \n";

deque<string>ss = {"a","b","c"};

stack<string>s(ss);

while(!s.empty()){

cout << s.top()<< " ";

s.pop();

}

cout <<"the tack does not support range iterator!"<< endl;

//--------------------queue----------------------------

cout << "\n5. int queue \n";

deque<string>s1 = {"a","b","c"};

queue<string>s2(s1);

while(!s2.empty()){

cout << s2.front()<< " ";

s2.pop();

}

//---------------------map------------------

}

