#include <iostream>

#include <string>

#include <vector>

using namespace std;

#ifndef PERSON\_H\_

#define PERSON\_H\_

class Person

{

public:

Person();

Person(int level, string name);

Person(int level, string name, string spouse);

virtual ~Person();

void have\_child(Person \*child);

void print() const;

private:

int level; // oldest ancestor is level 0

string name;

string spouse\_name;

Person \*parent;

vector<Person\*> children;

void print\_bar() const;

};

#endif /\* PERSON\_H\_ \*/

Person \*make\_tree();

Person::Person() :level(0){}

Person::Person(int level, string name) : level(level), name(name){}

Person::Person(int level, string name, string spouse) : level(level), name(name), spouse\_name(spouse){}

Person::~Person()

{

if(parent!=nullptr)

{

parent->children.erase(parent->children.begin() + 0);

}

}

void Person::have\_child(Person \*child)

{

children.push\_back(child);

child->parent = this;

}

void Person::print() const

{

if (name != ""&&spouse\_name != ""&&level==0)

{

cout << name <<" ("<< spouse\_name << ")" << endl;

print\_bar();

}

else if (name != ""&&spouse\_name == ""&&level==0)

{

cout << name << endl;

}

else if (name != ""&&spouse\_name != ""&&level==1)

{

cout << "+---"<<name << " (" << spouse\_name << ")" << endl;

}

else if (name != ""&&spouse\_name == ""&&level==1)

{

cout<<"|"<<endl;

cout <<"+---"<< name << endl;

cout<<"|"<<endl;

}

else if (name != ""&&spouse\_name != ""&&level==2)

{

cout<<" ";print\_bar();

cout <<" "<<"+---"<< name << " (" << spouse\_name << ")" << endl;

}

else if (name != ""&&spouse\_name == ""&&level==2)

{

if(spouse\_name=="")

{

cout<<"| ";print\_bar();

cout <<"| "<<"+---"<< name << endl;

}

else

{

cout <<" "<<"+---"<< name << endl;

}

}

else if (name != ""&&spouse\_name != ""&&level==3)

{

cout <<" "<<"+---"<< name << " (" << spouse\_name << ")" << endl;

}

else if (name != ""&&spouse\_name == ""&&level==3)

{

if((parent->parent->children.size()==1))

{

cout<<" |"<<endl;

cout <<" "<<"+---"<< name << endl;

}

else if((parent->parent->children.size()==2))

{

cout<<" | |"<<endl;

cout <<" | "<<"+---"<< name << endl;

}

}

if (children.size() != 0)

{

children[0]->print();

}

else

{

this->~Person();

if (parent->children.size() != 0)

{

parent->children[0]->print();

}

else

{

parent->~Person();

if (parent->parent->children.size() != 0)

{

parent->parent->children[0]->print();

}

}

}

}

void Person::print\_bar() const

{

cout<<"|"<<endl;;

}

Person \*make\_tree()

{

Person \*charles = new Person(0, "Charles", "Mary");

Person \*susan = new Person(1, "Susan", "Bob");

Person \*george = new Person(1, "George");

Person \*tom = new Person(1, "Tom", "Alice");

charles->have\_child(susan);

charles->have\_child(george);

charles->have\_child(tom);

Person \*dick = new Person(2, "Dick");

Person \*harry = new Person(2, "Harry");

susan->have\_child(dick);

susan->have\_child(harry);

Person \*eliza = new Person(2, "Eliza", "Bud");

Person \*charlotte = new Person(2, "Charlotte", "Frank");

Person \*emily = new Person(2, "Emily", "Carl");

tom->have\_child(eliza);

tom->have\_child(emily);

tom->have\_child(charlotte);

Person \*tim = new Person(3, "Tim");

emily->have\_child(tim);

Person \*carol = new Person(3, "Carol");

Person \*sara = new Person(3, "Sara");

charlotte->have\_child(carol);

charlotte->have\_child(sara);

return charles;

}

int main()

{

Person \*root = make\_tree();

root->print();

delete root;

cout << endl << "Done!" << endl;

return 0;

}