import java.util.function.Function;

// TIPO = tipo di dato degli elementi

public class GenSchemeList<TIPO> { //VARIABILI DI ISTANZA

private final boolean vuoto;

private final TIPO primoElemento;

private final GenSchemeList<TIPO> restoLista

//public GenSchemeList<TIPO> NULL\_LIST = new GenSchemeList<TIPO>();

public GenSchemeList(){ // COSTRUTTORE VUOTO

vuoto = true;

primoElemento = null;

restoLista = null;

}

// COSTRUTTORE CON PARAMETRI

public GenSchemeList(TIPO el, GenSchemeList<TIPO> L){

vuoto = false;

primoElemento = el;

restoLista = L;

}

public boolean isNull(){ return vuoto; }

public TIPO car(){ return primoElemento; }

public GenSchemeList<TIPO> cdr() { return restoLista; }

public GenSchemeList<TIPO> cons(TIPO el){ return new GenSchemeList<TIPO>(el,this); }

public int listLength(){

if (isNull()) return 0;

return 1+cdr().listLength();}

public TIPO listRef(int n){

if(n==0) return car();

public GenSchemeList<TIPO> reverse(){

if (isNull()) return this;

// lista nulla

GenSchemeList<TIPO> lista = new GenSchemeList<TIPO>();

// lista corrente

GenSchemeList<TIPO> resto = cdr();

lista = lista.cons(car());

while (!resto.isNull()) {

lista = lista.cons(resto.car());

resto = resto.cdr(); }

return lista; }

public String toString(){

String desc = "(";

if (!isNull()){

//CONTIENE CARATTERI E LI STAMPA IN STRINGA

desc = desc +(car()).toString();

GenSchemeList<TIPO> resto = cdr();

while (!resto.isNull()){

desc = desc + ","+ resto.car().toString() ;

// TOGLIE IL PRIMO ELEMENTO

resto = resto.cdr();

}

} return desc+")"; }

// APPLICA LA FUNZIONE f a tutti gli elementi della lista

public GenSchemeList<TIPO> map(Function<TIPO,Object> f){

if (isNull()) return new GenSchemeList<TIPO>();

return new GenSchemeList<TIPO>((TIPO)f.apply(car()),cdr().map(f)); }

public GenSchemeList<TIPO> append(GenSchemeList<TIPO> Lista){

if (isNull()) return Lista;

return (cdr().append(Lista)).cons(car()); }

}