Нововолинський ліцей-інтернат Волинської обласної ради

ТВОРЧИЙ ПРОЕКТ «ПЕРЕБОРНІ ЗАДАЧІ ПАСКАЛЬ»

Виконав:
учень 10-В класу
Лисиця Павло
Вчитель:
Шаповал Юрій В’ячеславович

Нововолинськ-2015

Зміст

1. Переборні задачі (перестановки, лексичний перебір, перебір з поверненням).

2. Жадні алгоритми, як підтип переборних задач.

3. Висновок.

4. Список використаної літератури.

КІНЕЦЬ

1. **Перебір**

(перестановки, лексичний перебір, перебір з поверненням)

Переборні задачі

Класичним прикладом переборної задачі служить задача комівояжера. Дано множини з N міст , відстань між якими відома. В якому порядку повинний проходити їх комівояжер , ходячи в кожне місто один раз , щоб пройдений шлях був найкоротший ? Скільки існує перестановок із N міст?

Почнемо з простіших задач.

Утворити всі можливі перестановки трьох цифр 1,2,3

для i від 1 до 3 пц

для j від 1 до 3 пц

для k від 1 до 3 пц

Задача №1

якщо(I <>j) і (j<>k) і (k<>I) тоді
вивести I,k,j
все
кц
кц
кц

2) Отримати всі перестановки чисел від 1,2,3….,n

поч
ввести масив A[1..n]
t:=0
виконати рекурсивну процедуру генерації
кін
Підпрограма генерації
поч
якщо t<n тоді
для j від t+1 до n пц
переставляємо елементи a[t+1], a[j]
збільшуємо t
виконуємо генерацію

зменшуємо t
переставляємо елементи a[t+1], a[j]
кц
якщо t=n тоді виводжу масив все
кін

3)Генеруємо перестановки, використовуючи множинний тип величи

const n=3;

Лексичний перебір

1.Повернемось до перебору:
а). Ми мали: 1,2,3
1,3,2
2,1,3
2,3,1
3,2,1
3,1,2
б). А якщо ми маємо
3,8,7
Як утворити всі можливі перестановки?
1. Утворити перестановки 1,2,3 і використати їх як індексний масив.
в) Маємо 1,1,2
1,2,1
2,1,1
2. Побудуємо лексичний перебір для довільних елементів масиву
X=3 2 4 2 4 3 1

а) Рухаємось справа наліво. Крок вперед можна зробити, якщо наступне число більше за попереднє. Ми зупинилися перед числом 2. Це число потрібно помітити
X=3 2 4 2 4 3 1
б) Рухаємось справа наліво. Крок вперед можна зробити, якщо число менше за знайдене

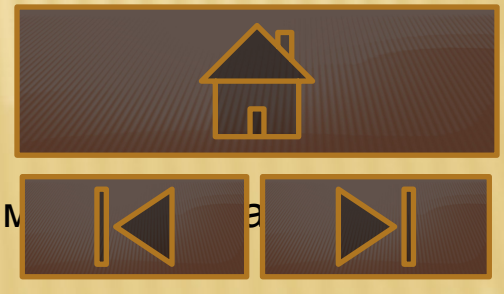
б). А якщо ми маємо
3,8,7
Як утворити всі можливі перестановки?

1. Утворити перестановки 1,2,3 і використати їх
2. як індексний масив.

в) Маємо 1,1,2
1,2,1
2,1,1

2. Побудуємо лексичний перебір
для довільних елементів масиву
X=3 2 4 2 4 3 1

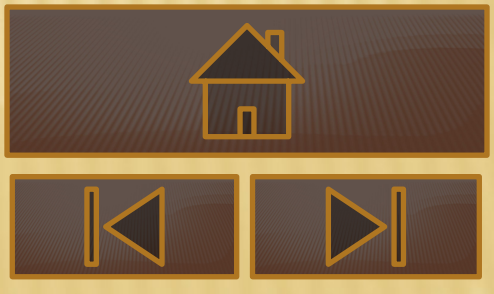
а) Рухаємось справа наліво. Крок вперед можна зробити, якщо наступне число більше за попереднє. Ми зупинилися перед числом 2. Це число потрібно помітити.
X=3 2 4 2 4 3 1



б) Рухаємось справа наліво. Крок вперед можна зробити, якщо число м[illegible]е число(2). Ми зупинилися перед числом 3. Це число потрібно помітити.

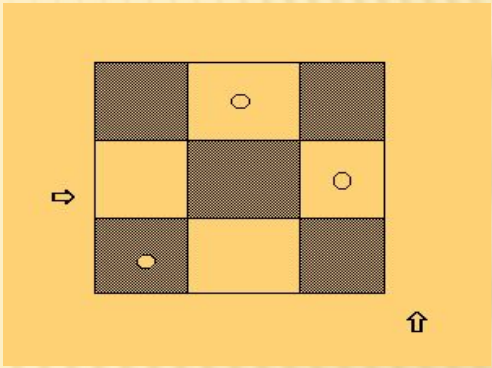
programlecsychny_pereb ir;

```
usescrt;
var n,i:integer;
        x:array [1..100] of integer;
functionnext:boolean;
vark,j,temp:integer;
        found:boolean;
        begin
i:=n;
found:=false;
while (not found)and(i>1)do
begin
found:=x[i-1]<x[i];
if(not found)then i:=i-1 else k:=i-1;
end;
if found then begin
i:=n;
while x[i]<=x[k] do i:=i-1;
temp:=x[i];
x[i]:=x[k];
x[k]:=temp;
i:=k+1;
j:=n;
```

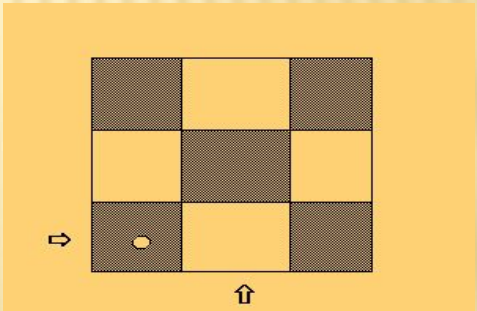


Перебір з поверненням

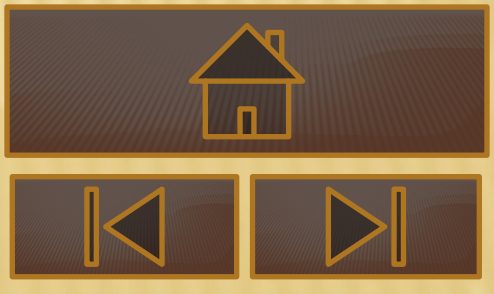
Розглянемо метод розв'язку цілого ряду переборних задач на прикладі відомої задачі про тури, які треба розставити на шахівниці так, щоб вони не били один одного. Для наочності візьмемо дошку 3х3 клітинки і розставляти будемо відповідно 3 тури. З відомих формул комбінаторики випливає, що ми будемо мати 3!=6 варіантів розміщень. Очевидно. що при будь-якім розміщенні на кожній горизонталі і на кожній вертикалі повинно бути по одній турі. При відомій вправності і фантазії 3 тури можна ще розставити "вручну" . Ну, а вісім чи більше (на відповідній дошці, звичайно)? Важкувато.Спробуємо перекласти цю задачу на плечі машини. Нехай ми маємо два покажчи*ки - стрілки*



Одна з них указує на горизонталь дошки, інша - на вертикаль. Ставимо першу туру і покажчики, як показано на малюнку, і переміщаємо горизонтальний покажчик на одну позицію вправо. Пробуємо ставити в клітинку, на яку вказують покажчики. Але це зробити не можна. Піднімаємо вертикальний покажчик на одну позицію нагору. Клітка, на яку вказують покажчики, не бита, можна ставити туру. Переміщаємо горизонтальний покажчик на одну клітинку вправо. Знову пробуємо поставили туру туди, куди вказує покажчик, але клітка бита.



Піднімаємо вертикальний покажчик
на одну позицію вгору. Клітка вільна, ставимо туру, горизонтальний покажчик вийде за межі дошки. Це ознака того, що дане розміщення довершене. Виводимо результат і повертаємо горизонтальний покажчик на одну позицію вліво, вертикальний покажчик установлюємо на туру в даній вертикалі і намагаємося підняти туру, на яку він указує . Вільних кліток немає. Знімаємо туру, горизонтальний покажчик уліво на одну позицію, а вертикальний поміщаємо на ту горизонталь, де є тура в даній вертикалі. Намагаємося знову підняти туру, на яку вказує покажчик. Це можливо. Піднімаємо її і горизонтальний покажчик на 1 позицію вправо, а вертикальний - на 1. Пробуємо помістити туру по покажчиках, якщо немає - піднімаємо вертикальний наверх, поки не знайдемо не биту клітку. Ставимо туру, і знову горизонтальний покажчик піде на одну позицію вправо. Готове чергове розміщення. Знову після виведення результату повернемо горизонтальний покажчик на одну позицію вправо , а вертикальний установимо на туру в цій вертикалі і спробуємо підняти туру, на яку він указує. Вільних кліток немає. Знімаємо туру і, перемістивши горизонтальний покажчик ще раз вправо , вертикальний...



...виставляємо проти тури у відповідній вертикалі і намагаємося підняти її. У нас знову нічого не вийде, ми знімаємо і цю туру і переміщаємо горизонтальний покажчик ще лівіше, а вертикальний - на туру в стовпці, на який вказує горизонтальний покажчик. Пробуємо підняти її. Це зробити вдається. Повторюємо всі ці дії , при цьому, якщо горизонтальний покажчик виходить за межі дошки вправо, виводимо розміщення, а ознакою того, що всі комбінації вже були, стане те, що горизонтальний покажчик вийде за межі дошки вліво.

Виберемо структуру даних.

Поле представимо у виді матриці A[n:n], де N кількість кліток у кожній горизонталі і вертикалі (та й тур у нашому випадку теж N). Якщо в даній клітинці немає тури - A[i,j]=0 , а якщо є то A[i,j]=1. Ще нам знадобляться дві змінні цілого типу для збереження в них значення покажчиків П_В и П_Г.

Для конструювання алгоритму на високому рівні деталізації нам необхідно мати такі процедури і функції:

ПОСТАВ_І_ВПРАВО - ставить туру в клітинку з заданими координатами і пере
покажчик на одну позицію вправо , а вертикальний установлює на першу позицію
ЗНІМИ_І_ВЛІВО - знімає туру з даної клітки і переміщає горизонтальний покажч
вертикальний покажчик встановлює в позицію тури, на яку тепер указує горизонтальний покажчик (
процедура)

ПРАВИЛЬНА_КЛІТИНКА - логічна функція. Повертає ***істина***, якщо туру можна
хибно, якщо поставити в клітинку не можна

Тепер наш алгоритм може бути представлений так:

Пошук_з_поверненням

Алг.

поч

для i **від** 1 **до** n

нц

для j **від** **1** **до** n

нц

A[i,j] :=0

кц

кц

Як працюють процедури і функції , використовувані основним алгоритмом, зрозуміло з Pas
- реалізації алгоритму.

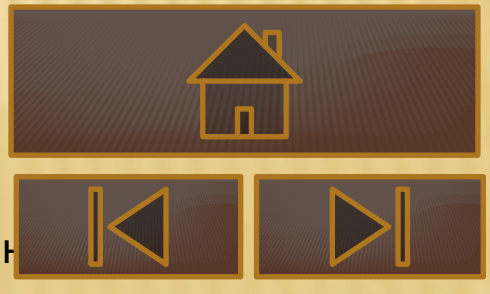
```
usescrt;
const n=3;
var i,j,k,y_v,y_g:longint;
 a:array[0..n+1,0..n+1] ofinteger;
 f:text;
procedure pr1;
begin
      a[y_v,y_g]:=1;
      y_g:=y_g+1;
      y_v:=1;
end;
procedure pr2;
begin
for i:=1 to n do a[i,y_g]:=0;
    y_g:=y_g-1;
for i:=1 to n do
if a[i,y_g]=1 then y_v:=i;
                a[y_v,y_g]:=0;
                y_v:=y_v+1; end;
```

```
functionpr:boolean;
begin
pr:=true;
for i:=1 to n do
if (a[y_v,i]=1) thenpr:=false;
end;
begin
clrscr;
 k:=0;
for i:=1 to n do
for j:=1 to n do
 a[i,j]:=0;
 y_v:=1;
 y_g:=1;
 pr1;
while y_g<>0 dobegin
while (not(pr)) and (y_v<n+1) do y_v:=y_v+1;
if y_v<n+1 then pr1 else pr2;
ify_g=n+1 thenbegin pr4;pr3; end;
end;
end.
```

Якщо для тур ми перевіряли горизонталі, то з ферзями прийдеться потурбуватися і про діагоналі. Функція буде мати вид:

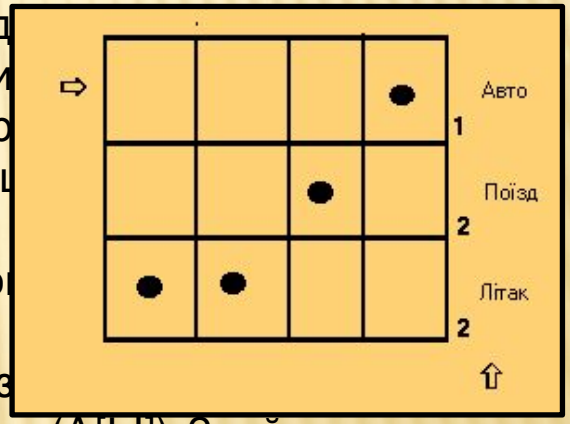
```
Functionpr:boolean;{ чи можна ставити}
Var c: boolean;
    m: integer;
Begin
  m:=1;
  c:=True;                                    {можна!}
if y>n then c:=false
else
While (m<=n) and (c=True) do
```

```
begin
if (a[m,y]<>0)
then c:=False;                    {не можна, горизонталь зайн
    m:=m+1;
```



Метод може бути використаний і в тому випадку, якщо потрібно одержати комбінації об'єктів, частина з яких однотипні.

Для того, щоб перебороти шлях від початкового пункту до кінцевого, потрібно пройти чотири ділянки маршруту. Кожний з ділянок можна перебороти або літаком, або потягом, або автомобілем. Літаком і потягом можна скористатися двічі, а автомобілем - тільки один раз. Потрібно вказати усі варіанти подолання шляху. Складіть програму, що виводить на екран усі варіанти подолання шляху від початкового пункту д



“Ігрове поле “ стало абстрактним, по вертикалі в нас тепер види горизонталі - номер прохідної ділянки маршруту Матриця тепер види транспорту і 4 ділянки шляху) . Тепер у тих горизонталях, потягу, можна ставити не по однієї фішці, а по дві.

На малюнку представлене положення покажчиків і фішок до мо другої послідовності видів транспорту на маршруті.

Аналогічну задачу можна розв’язати і на виготовлення виробу з таблиці A[1..N,1..N] занесено час виготовлення J деталі на I станку (A[I,J]). Знайти мінімальний час виготовлення виробу, якщо всі деталі починають виготовляти одночасно.

{Виріб складається з N деталей, кожна з яких може вироблятися на довільному з N станків. Час виготовлення **j** деталі на **i** станку міститься в таблиці **T[i,j].**

Виготовленнявиробупочинається на всіх станках одночасно. Знайти мінімальни час виготовити виробу, якщо всі деталі починають виготовляти одночасно.

Вхідні дані в файлі DETAL.DAT:
3
3 2 7
1 3 2
5 6 2
Вихідні дані в файлі DETAL.REZ:
2
2 1 3}

```
program DETAL1;
usescrt;
var min,n,i,j,k,y_v,y_g,max:longint;
t,a:array[0..100,0..100] of integer;
time:array[1..100] of integer;
f:text;
procedure pr1;
begin
a[y_v,y_g]:=1;
y_g:=y_g+1;
y_v:=1;
end;
```

```
begin
pr:=true;
fori:=1 to n do
if (a[y_v,i]=1) then pr:=false;
end;
begin
assign(f,'detal.dat');
reset(f);
readln(f,n);
fori:=1 to n do
for j:=1 to n do read(f,t[i,j]);
close(f);
clrscr;
k:=0;
min:=maxint;
for i:=1 to n do
for j:=1 to n do
a[i,j]:=0;
y_v:=1;
y_g:=1;
 pr1;
```

Жадні алгоритми

Жадний алгоритм-це алгоритм, який накожному кроці робить локально найкращий вибір в надії, що результат буде оптимальний.

Задача1.

Написати програму, яка розмінює деяку суму грошей найменшим числом купюр. (Маємо купюри номіналом 1грн, 2грн, 5грн, 10грн, 20грн, 50грн, 100грн, 200грн, 500грн, 1000грн, 5000грн).

Задача2.

Марсіани Міша і Маша вирішили разом підібрати подарунок на день народження Каті. Коли вони нарешті знайшли те, що хотіли, і упакували предмет в красиву коробку, потрібно було вирішити, як підписати подарунок. Друзі подумали, що кращим рішенням буде скласти загальний підпис так, щоб в ній як підрядки містилися їх імена. Врахуйте, що на Марсі прийнято підписуватися повними іменами, а вони у марсіан можуть бути досить довгими.

Вхідні дані.

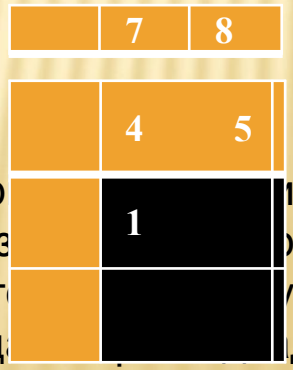
Вхідний файл INPUT.TXT містить два рядки, в яких записані повні імена друзів. Імена, як недивно, складаються з букв латинського алфавіту, з яких тільки перша, -прописна. Довжина імен не перевершує1000.

Вихідні дані.

У вихідний файл OUTPUT.TXT виведіть найкоротший рядок, в якому зустрічаються імена Миші і Маша одночасно. Букви, з яких імена починаються в цьому рядку треба зробити великими. Якщо існує декілька рішень, виведіть те, яке менше в алфавітному порядку.

Динамічне програмування

Динамічним програмування називають метод ,який дозволяє розв'язувати “переборні” задачі, спираючись на вже розв'язані під задачі меншого розміру. Прицьому необхідно, щоб всі розв'язки можна було заповнити в таблиці (тобто дані, обчислені один раз,не обчислюються знову).
Ідеї динамічного програмування сильно подібні на метод математичної індукції.
Сформулюємо необхідні вимоги:



1.Всі розв'язки під задач повинні збер ці.
2.Існує відомий розв'язок для задачі з ністю (аналогічно мінімального параметра в методі мат кції).
3.Існує спосіб виразити розв'язок зад дачі (можливо, від задачі) меншої розмірності. Це більше всього нагадує рекурентнес пів математиці.

Напишіть програму, що визначає кількість телефонних номерів довжини N, набираються ходом коня.
Вхідні дані. У вхідному файлі записано ціле число N(1 ≤ N ≤ 20).
Вихідні дані. Виведіть у вихідний файл шукану кількість телефонних номерів.
Приклад вхідних та вихідних даних

input.txt	output.txt
1	8
2	16
3	36

Задача2. Покупка білетів

За квитками на прем'єру нового мюзиклу вишикувалася черга з людей, кожен з яких хоче купити 1 квиток. На усю чергу працювала тільки одна каса, тому продаж квитків йшов дуже повільно, приводячи "постояльців" черги у відчай. Найкмітливіші швидко помітили, що, як правило, декілька квитків в одні руки касир продає швидше, ніж коли ці ж квитки продаються по одному. Тому вони
запропонували декільком людям, що підряд стоять, віддавати гроші першому з них, щоб він купив квитки на усіх.
Проте для боротьби із спекулянтами касир продавала не більше 3-х квитків в одні руки, тому домовитися таким чином між собою могли ше 2 або 3 людей, що стояли поряд.
Відомо,що на продажі-ій людині з черги одного квитка касир витрачає A і секунд, на продаж двох квитків – B і секунд, трьох квитків- C і секунд. Напишіть програму ,яка підрахує мінімальний час,за який могли бути обслужені усі покупці.
Зверніть увагу, що квитки на групу людей, що об'єдналися, завжди куп
ніхто в цілях прискорення некупує зайвих квитків (тобто квитків, які ніко

Приклад

Формат вхідних даних. У вхідному файлі записано спочатку число N- кількість покупців в черзі (1≤*N*≤5000). Далі йде N трійок натуральних чисел Ai, Bi, Ci. Кожне з цих чисел не перевищує 3600. Люди в черзі нумеруються починаючи від каси.

Формат вихідних даних. У вихідний файл виведіть одно число- мінімальний час в секундах, за який могли бути обслужені усі

b.in	b.out
5 51015 21015 555 20201 2011	12

Задача3. Задача про найбільшу зростаючу під послідовність.

Дано послідовність. Потрібно знайти довжину найбільшої зростаючої підпослідовності.
Вхідні дані.У першому рядку вхідного файлу записано число N- довжина послідовності (1≤N≤1000). У другому рядку записана сама послідовність (черезпробіл). Числа послідовності – цілі числа, не перевищують 10000 по модулю.
Вихідні дані. У вихідний файл потрібно вивести

p.in	p.out
6 32955286	3

послідовності.
Приклад

Задача 4. Представлення числа
Московская олимпиада по информатике 2005/06г. Олимпиада 10-11 классов, 1тур, 12 февраля 2006 года
Вчителька математики попросила школярів скласти арифметичний вираз, так щоб його значення дорівнювало числу N, і записати його в зошиті. У виразі можуть бути використані натуральні числа, небільші K, операції додавання і множення, а також дужки. Петя дуже не любить писати, і хоче придумати вираз, що містить якомога менше символів. Напишіть програму, яка допоможе йому в цьому.
Формат вхідних даних. У першому рядку вхідного файлу записано два натуральні числа :N (1≤N≤10000) – значення вираження і K (1≤K≤10000)- найбільше число, яке дозволяється використати у виразі.
Формат вихідних даних. У єдиному рядку вихідного файлу виведіть вираз з цим значенням, що записується найменшою можливою

Примітка. При підрахунку довжини вираження враховуються усі символи: цифри, знаки операцій, дужки.

ch.in	ch.out	.Пояс
73	3+1+3	5
1520	15	2
1761	(1+1+1+1)*(1+1+1+1)*(1+1+(1+1+1)*(1+1+1))	41

Висновок

Трудність переборних задач в тому, що кількість передбачуваних рішень буває дуже велика (необмежена).

Для 50 міст, якби комп`ютер виконував по 1000000 (млн.) перестановок в секунду (поки ні), то опрацював би за час існування Всесвіту.

Як виходити з цієї ситуації**?** Відкидати наперед неправильні рішення: якщо початок наступного перебору більший за якийсь мінімальної довжини прохід, то зразу відкинути йог і всі подальші, котрі містять таку підпослідовність.

Список використаної літератури:

www.slavdpu.dn.ua/fmk/publications/manuals/met_olimp_last.pdf.

ukped.com/.../2555-prikladni-zadachi-ta-yih-matematichni-modeli.html

pascal.proweb.kz/index.php?page=117

distance.edu.vn.ua/metodic/pascal/2.htm