Міністерство освіти і науки України
Національний технічний університет України «КПІ ім. Ігоря Сікорського»
Факультет інформатики та обчислювальної техніки

Кафедра інформатики та програмної інженерії

Мультипарадигменне програмування

ЗВІТ

до лабораторної роботи №5

Виконав студент ІП-01 Пасальський Олександр _____
(№ групи, прізвище, ім'я, по батькові)

Прийняв ас. Очеретяний О. К. _____
(посада, прізвище, ім'я, по батькові)

Київ 2022

Завдання:

Exercise 2.1

мапа налічує n регіонів, підрахуйте максимальну кількість обчислень які необхідно зробити для визначення чи є конфлікт кольорів? Обгрунтуйте вашу відповідь із використанням дерева фактів програмного коду

Exercise 2.2.1

Використовуючи першу програму факторіал покажіть що не існує дерева фактів що має корінь 'factorial(5,2)' що має всі листки true

Exercise 2.2.2

Напишіть дерево фактів для цілі 'factorial(3,1,6)' маючи всі листки true, в такому ж форматі як це зроблено для factorial(3,6) попередньо. Як ці дві програми відрізняються з точки зору підрахунку факторіалу? Також виконайте трасування цілі 'factorial(3,1,6)' використовуючи Prolog

Exercise 2.3.1

Напишіть дерево фактів для цілі 'move(3,left,right,center) , покажіть що це програмна послідовність.

Яким чином це дерево фактів стосується процесу підстановки описаного вище.

Exercise 2.3.2

Виконайте ціль Prolog move(3,left,right,left). У чому помилка? Запропонуйте шлях вирішення та перевірте що виправлення спрацювало.

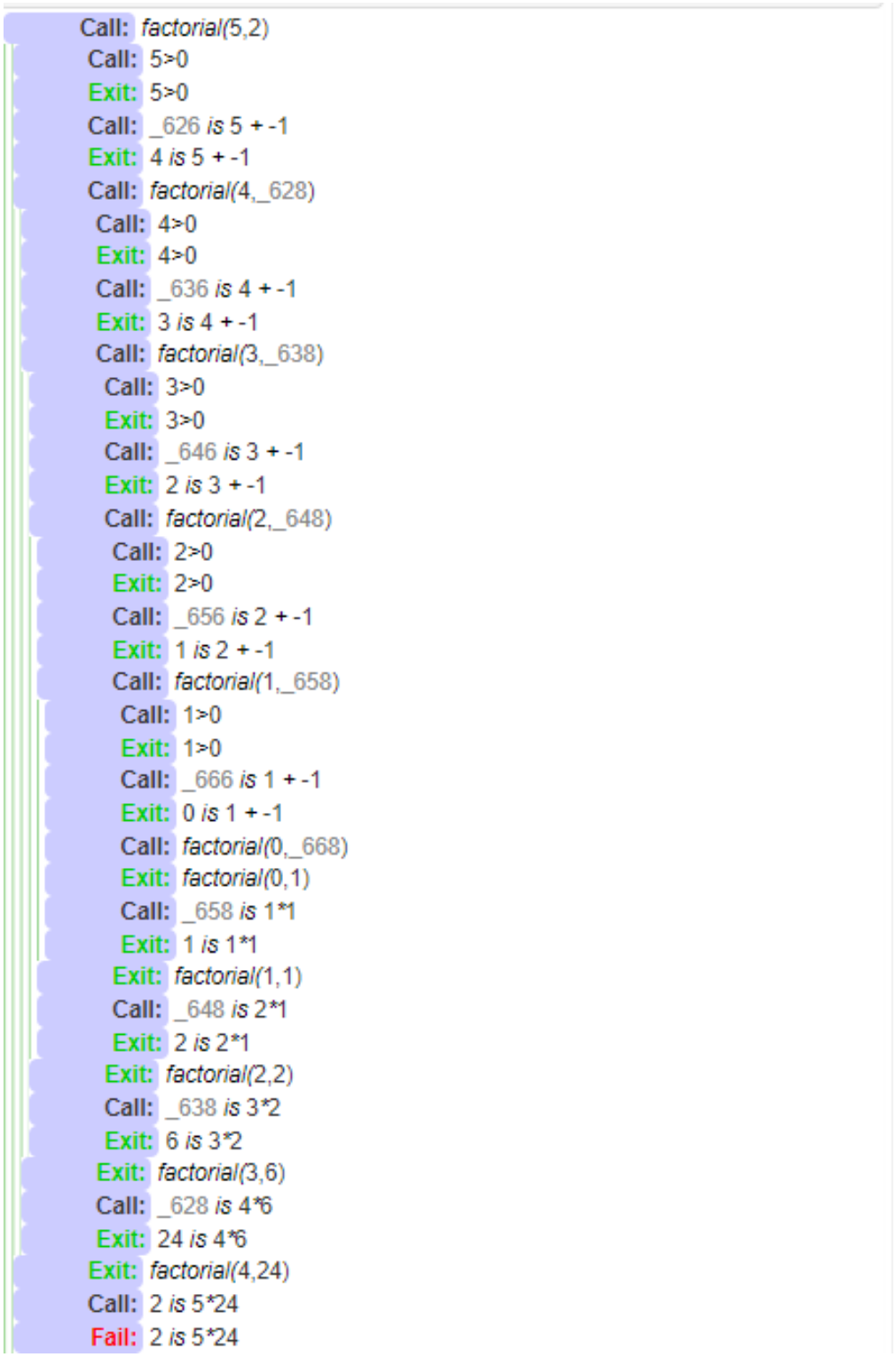
Розв’язання:

Exercise 2.1

Задача на перевірку конфліктів кольорів на карті зводиться до перебору всіх пар вершин (перевірки чи колір не співпадає з сусідом) графа, де вершини це регіони, а ребра між вершинами є якщо відповідні регіони межують. Граф має n вершин і в найгіршому випадку кожна вершина зв’язана з n-1 вершинами отже кількість операцій n*(n-1), проте в даному варіанті кожна пара буде перевірена двічі. Тому нам потрібно перебрати n * (n-1) / 2 пар вершин. Для кожної такої пари вершин потрібно перевірити: чи вершини суміжні і чи їхній колір не співпадає, складність таких перевірок складе константу 3. Отже, максимальна кількість обчислень які необхідно зробити для визначення чи є конфлікт кольорів становить 3 * n * (n-1) / 2

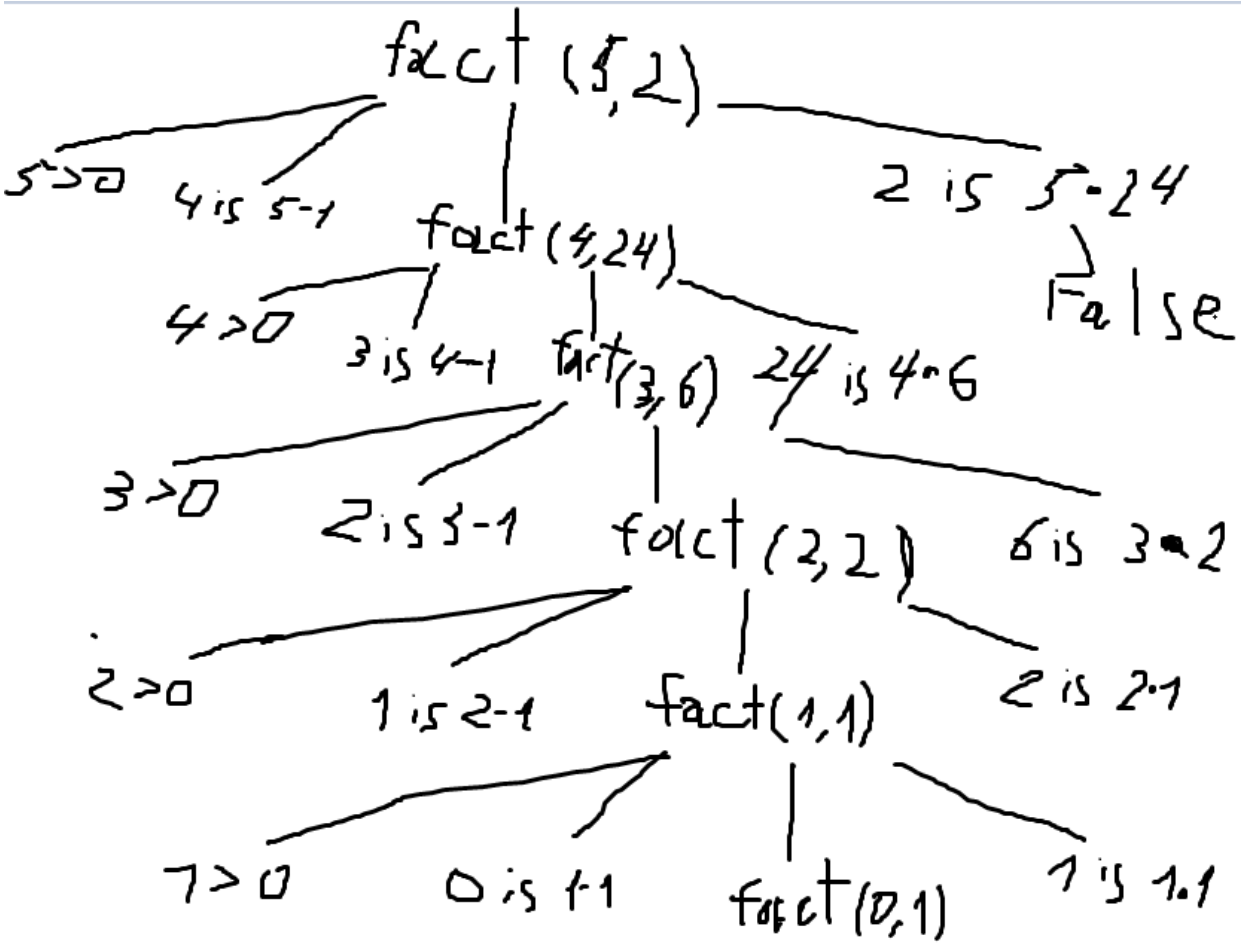
Exercise 2.2.1

Поглянемо на обхід дерева використовуючи Prolog, виконаєм трасування для 'factorial(5,2)'



Бачимо, що хибний результат отримано в листі де перевіряється чи дане число рівне знайденому факторіалу (2 == 5*24) отримали в листі False і вся функція повернула False.

Побудуємо дерево даного трасування.

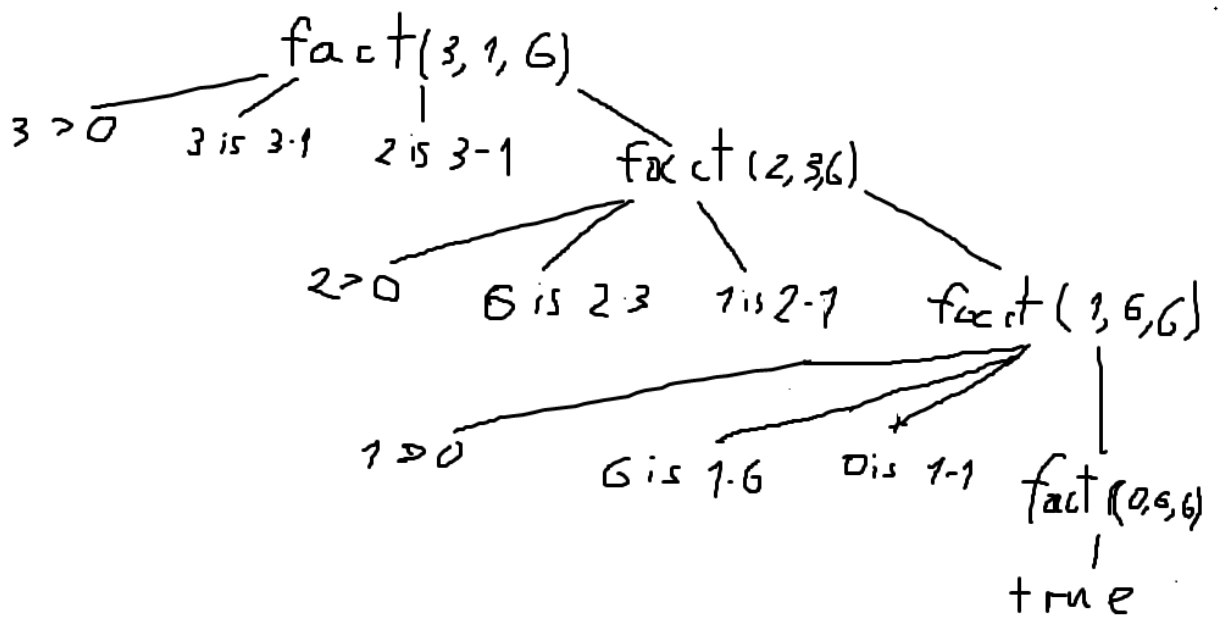


Exercise 2.2.2

Поглянемо на обхід дерева використовуючи Prolog, виконаєм трасування для 'factorial(3, 1, 6)'

```
Call: factorial(3,1,6)
 Call: 3>0
 Exit: 3>0
 Call: _624 is 3*1
 Exit: 3 is 3*1
 Call: _638 is 3 + -1
 Exit: 2 is 3 + -1
 Call: factorial(2,3,6)
  Call: 2>0
  Exit: 2>0
  Call: _640 is 2*3
  Exit: 6 is 2*3
  Call: _654 is 2 + -1
  Exit: 1 is 2 + -1
  Call: factorial(1,6,6)
   Call: 1>0
   Exit: 1>0
   Call: _656 is 1*6
   Exit: 6 is 1*6
   Call: _670 is 1 + -1
   Exit: 0 is 1 + -1
   Call: factorial(0,6,6)
   Exit: factorial(0,6,6)
  Exit: factorial(1,6,6)
 Exit: factorial(2,3,6)
Exit: factorial(3,1,6)
true
```

Побудуємо дерево



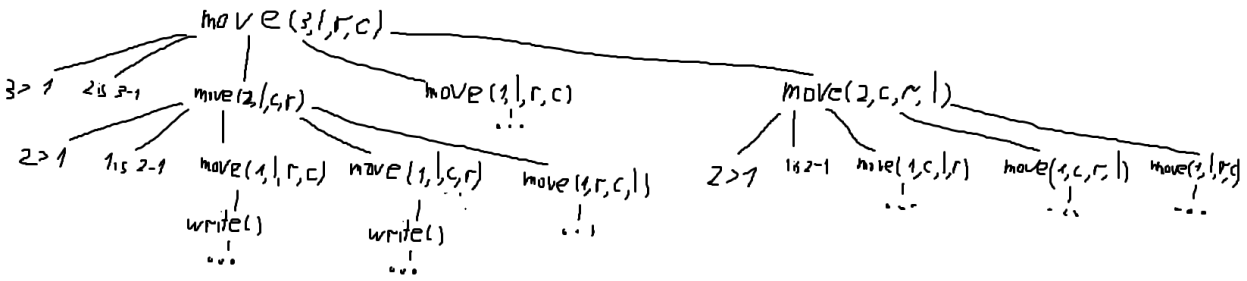
Дані дві програми відрізняються таким чином. Перша програма порівнювала обрахований факторіал з даним біля самого кореня, перед цим заглиблюючись рекурсивно до листків, проводячи необхідні обрахунки. Друга програма виконувала всі обрахунки до заглиблення (рекурсії), та перевіряла заданий та обчислений факторіал в листку, а наверх передавала результат в нашому випадку True.

Exercise 2.3.1

Поглянемо на трасування

Побудуємо дерево. Позначення:

l – left, r – right, c – center.



Повне дерево будувати нема сенсу, так як воно велике, а суть процесу можна зрозуміти і на скороченій версії дерева.

Exercise 2.3.2

Результат виконання move(3, left, right, left).

```
move(3,left,right,left).
Move top disk from left to right
Move top disk from left to left
Move top disk from right to left
Move top disk from left to right
Move top disk from left to left
Move top disk from left to right
Move top disk from left to right
```

Програма відпрацювала без помилок, проте має неоднозначний результат, так як вона говорить нам перемістити диск з лівої осі до лівої. По суті в нас в програмі є два різних “лівих” (на вхід ми передаєм двом різним осям однакові назви), через що розв’язок неточний і неприйнятний. Для вирішення цієї проблеми, в програмі потрібно перевіряти, чи на вхід передаються різні назви осей. Модифікуємо функцію, щоб вирішити цю проблему.

```
move(N,X,Y,Z) :-
 N>1,
 X\=Y, Y\=Z, x\=Z,
 M is N-1,
 move(M,X,Z,Y),
 move(1,X,Y,_),
 move(M,Z,Y,X).
```

Добавлений рядок перевірки осей, я підкреслив червоним. Перевіряючи осі, якщо є однакові повертаємо False.

Результат виконання

```
move(3,left,right,left).
false
```