Дубина Андрій

ІПС-21

12 варіант

1. Нумерація n-oк натуральних чисел. Основні тотожності.

Нумерація Кантора – нумерація, що задається наступним чином: всі пари натуральних чисел розташовуються у послідовність <0, 0>, <0, 1>, <1, 0>, <0, 2>, <1, 1>, <2, 0>, <0, 3>, <1, 2>, …

Числа тут впорядковані таким чином, що пара <x,y> йде раніше за <u, v> при x+y < u+v. Якщо x+y = u+v, тоді ставиться умова x<u. Кожна така пара має свій номер n, тобто існує бієкція між множиною нмоерів та множиною пар.

Нумерація Кантора позначається c(x,y) – номер пари <x,y> в множині пар. Також існують ліва та права координатні функції, що позначають відповідно l(n) та r(n) відповідно. Вони повертають вілповідно ліву або праву координату з пари під номером n.

З такого означення випливає наступна тотожність:

***c***(***l***(*n*), ***r***(*n*)) = *n*, ***l***(***c***(*x*, *y*)) = *x*, ***r***(***c***(*x*, *y*)) = *y*.

Функція c(x,y), а також ліва та права координатні функції є ПРФ. Це можна довести знаходженням алгоритму, що їх обчислює. Для c(x,y) це:

 function ***с***(*x*, *y*)

                               begin

*s* = 0

                                      for *i* = 0 to (*x* + *y*)

*s* = *s* + *i*

                                      for *i* = 0 to (*x* + *y*)

{*j =* (*x + y*)∸ *i*

                                              if  *x = i* ∧ *y* = *j* then *k* = *i*}

***с* =** *s* + *k*

                                end.

Для l(n) це:

 function ***l***(*n*)

                                begin

                                      for *i* = 0 to *n*

                                      for *j* = 0 to *n*

                                               if ***с***(*i*, *j*) = *n*  then ***l*** = *i*

                                 end

Для r(n) це буде аналогічний алгоритм, але замість i повертатиметься j.

2. Проблема зупинки алгоритмічно нерозв’язна*.* Довести.

Проблема зупинки є проблемою теорії алгоритмів, що заключається у неможливості визначити, чи завершиться виконання алгоритму при поданні скінченної кількості вхідних даних, чи він нескінченно працюватиме.

Доводиться ця проблема методом «від супротивного».

Припустимо, вже існує рішення проблеми – функція F(x, x), яка відповідає на питання: «чи зупиниться функція (перший аргумент) при роботі з даними (другим аргументом)?»

Можемо створити функцію U(x):

U(x)   
begin  
 i = 0  
 if(F(x,x) = true) then   
 while(true)  
 i = i+1

end

Позначимо запис такої функції як p. При виклику функції F(p,p) можливі два випадки:   
1. Якщо U зупиняється, то F(U(x),U(x)) повертає результат true. Але в такому випадку функція U(U(x)) не зупиниться, адже по запису в нас почнеться нескінченний цикл.  
2. Якщо U(x) не зупиняється, то F(U(x), U(x)) повертає false і в такому випадку функція має завершити роботу (end). Маємо протиріччя в обох випадках. Це протиріччя пояснюється нерозв’язністю проблеми зупинки, адже такої функції F(x,x), яка б розв’язувала нашу задачу, не існує. Доведено.

3. Нехай задані клінівські номери функцій f(x) і g(x). Знайти клінівський номер їх

суперпозиції.

Нехай f(x) = K(n, x), g(x) = K(m,x), n та m – клінівські номери для відповідних функцій.   
f(g(x)) = K(n, K(n,x)) =   
Тому клінівський номер для f(g(x)) = [a,n,m] при n та m – клінівські номери для f(x) та g(x) відповідно.