МАШИНА ПОСТА.

Повний опис класу частково рекурсивних функцій разом з тезою Чёрча, які надані в розділі 3, дає один з можливих розв'язків задачі про уточнення поняття алгоритму. Зрозуміло, що це розв'язання не цілком пряме, оскільки поняття обчислюваної функції є вторинним по відношенню до поняття алгоритму. Виникає питання: чи не можна уточнити саме поняття алгоритму, а потім з його допомогою визначити і клас обчислюваних функцій?

Відповідь на це питання була дана в 1936-1937 роках Е. Постом і А. Тьюрингом незалежно один від одного і майже одночасно з роботами А. Чёрча і С. Кліні. Основна думка Поста і Тюрінга полягала в тому, що алгоритмічні процеси – це процеси, які може виконувати відповідним чином сконструйована «машина». За допомогою точних математичних термінів цими ученими були описані досить вузькі класи таких машин, на яких можна було виконувати або імітувати всі алгоритмічні процеси, які фактично коли-небудь описувалися в математиці. Таким чином, зрозуміло, що машина Поста не є реальним обчислювальним пристроєм, а також як і її близький аналог – машина Тюрінга, зображує гіпотетичну конструкцією для математичного дослідження проблем теорії алгоритмів.

Розглянемо тепер алгоритмічну систему, яка зображена машиною Поста.

Конструкція машини Поста виключно проста. Вона складається з:

1. *Інформаційної нескінченної стрічки*, яка зображує необмежену пам'ять машини. Ця стрічка розділена на *секції* (*комірки*) однакового розміру. В кожній секції стрічки або нічого не записано (така секція називається *порожньою*), або записана мітка $\vee$ (тоді секція називається *позначеною*). При цьому будемо вважати, що система координат жорстко зв'язана із стрічкою, а секції занумеровані цілими числами від $-\infty$ до $+\infty$. Інформація про те, які секції порожні, а які позначені, утворюють *стан стрічки*. На рисунку 5.1. зображено деякий стан інформаційної стрічки машини Поста.

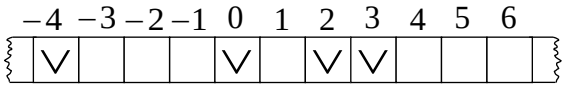


Рисунок 5.1 – Стан стрічки машини Поста

Іншими словами, стан стрічки – це функція, яка кожному числу (номеру секції) ставить у відповідність або мітку, або порожнє слово.

2. *Каретки*, яка інакше називається *головкою читання та запису*. Каретка може пересуватися уздовж стрічки вправо і вліво. Коли каретка нерухома, то вона стоїть напроти рівно однієї секції стрічки і тоді говорять, що каретка *оглядає* цю секцію, або *тримає її в полі зору*.

Введемо поняття *стан машини Поста*. Він складається із стану стрічки і вказівки номера тієї секції, яку оглядає каретка. На рисунку 5.2 зображений деякий стан машини Поста.

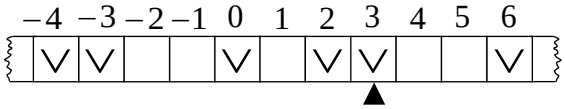


Рисунок 5.2 – Стан машини Поста

За одиницю умовного часу (за крок) стан машини Поста може змінитися: каретка може здвинутися на одну секцію вліво або вправо; каретка може надрукувати або стерти мітку в тій секції, напроти якої вона стоїть; каретка може розпізнати стоїть чи ні мітка в секції, яку вона тримає в полі зору.

Робота машини Поста якраз в тому і полягає, що каретка пересувається уздовж

інформаційної стрічки і друкує або стирає мітки. Ця робота відбувається за певною інструкцією – за *програмою*. Кожна програма складається з *команд*. Існує всього шість видів команд для машини Поста.

1. Команди руху вправо: $i.\ \Rightarrow\ j.$
2. Команди руху вліво: $i.\ \Leftarrow\ j.$
3. Команди друку міток: $i.\ \vee\ j.$
4. Команди стирання міток: $i.\ \xi\ j.$
5. Команди передачі управління: $i.\ ?(j_1,\ j_2).$
6. Команди зупинки: $i.$ *стоп.*

Символи i, j, j_1 та j_2 – це усюди натуральні числа $1, 2, 3, \ldots, k$; i – це *номер команди*; j *– відсилка*; j_1 *– верхня відсилка*; j_2 *– нижня відсилка*.

Таким чином, програма машини Поста зображує скінченний непорожній список команд машини Поста, який має дві властивості:

1. Список команд нумерується від 1 послідовно до k.

2. Відсилка будь-якої з команд списку повинна співпадати з номером деякої існуючої команди списку.

Розглянемо приклади правильно і неправильно записаних (з погляду синтаксису, а не змісту) програм машини Поста.

1. *стоп.* 2. $?(4, 1).$ 3. ξ 3. 4. *стоп.*	*вірно записана програма*

;

2. $?(4, 1).$ 1. *стоп.* 3. ξ 3. 4. *стоп.*	*невиконана перша умова*

;

1. *стоп.* 2. $?(4, 5).$ 3. ξ 3. 4. *стоп.*	*невиконана друга умова*

Для того щоб машина Поста почала робити, треба, по-перше, задати деяку програму, а по-друге, деякий стан машини. Для простоти будемо вважати, що в *початковому стані* машини каретка завжди ставиться проти секції з номером нуль. При такій угоді стан машини повністю визначається станом стрічки.

Робота машини згідно заданої програми при заданому початковому стані відбувається таким чином. Машина виконує першу команду за один крок. Потім за один крок виконується команда, номер якої рівний значенню відсилки і так далі. Якщо на деякому k-ому кроці виконалася команда, яка не має відсилки ($k.$ *стоп.*), то на $(k+1)$-ому кроці не виконується ніяка команда, тобто машина зупиняється.

Тепер залишилося пояснити, як відбувається виконання всіх шести команд машини Поста.

Перша команда: Каретка здвигається на одну секцію вправо.

Друга команда: Каретка здвигається на одну секцію вліво.

Третя команда: Каретка ставить мітку в секції, яка оглядається, в тому випадку, якщо перед виконанням команди секція порожня. В іншому випадку команда вважається нездійсненною.

Четверта команда: Каретка знищує мітку в секції, яка оглядається, в тому випадку, якщо ця секція позначена. В іншому випадку команда вважається нездійсненною.

П'ята команда: Виконання цієї команди не змінює стан машини (жодна з міток не знищується і не ставиться, каретка також залишається нерухомою). Якщо секція, яка оглядається перед початком виконання команди, була порожньою, то наступною повинна виконатися команда з номером j_1. Якщо секція, яка оглядається, позначена, то виконується відсилка на команду програми з номером j_2.

Шоста команда: Виконання цієї команди також не змінює стан машини Поста і полягає в тому, що машина зупиняється.

Таким чином, якщо задатися програмою і яким-небудь початковим станом, пустити машину в хід, то можливо здійснення одного з трьох наступних варіантів:

1. *Безрезультативна зупинка* – коли в ході виконання програми машина дійде до виконання нездійсненної команди (друкування мітки в непорожній секції; стирання мітки в порожній секції; передача управління на команду з неіснуючим номером).

2. *Результативна зупинка* – коли в ході виконання програми машина дійде до виконання команди *«стоп»*. Програма в цьому випадку вважається виконаною.

3. *Нескінченна робота* – коли в ході виконання програми машина не дійде до виконання жодній з команд, які призводять до результативної або безрезультативної зупинки.

Як приклад, розглянемо роботу машини Поста, яка задана програмою і початковим станом, зображеними на рисунку 5.3.

$1.\vee 4. \quad 3.\Leftarrow 2. \quad 5.?(4,3).$

$2.\xi 3. \quad 4.\Rightarrow 5.$

Рисунок 5.3 – Стан і програма машини Поста

Легко бачити, що кожний крок цієї роботи породжує ланцюжок станів машини Поста, зображений на рисунку 5.4.

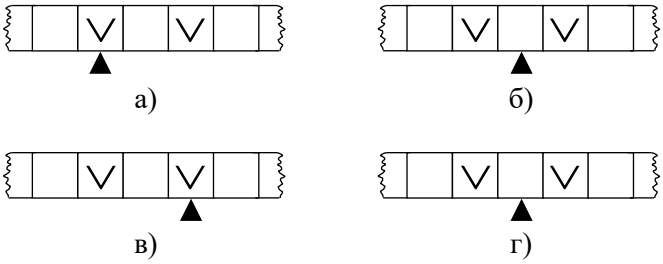


Рисунок 5.4

Після виконання команди 1 в секцію, яка оглядається кареткою, записується мітка, а стан машини Поста змінюється так, як це наведено на рисунку 5.4 а). Протягом другого кроку виконується команда з номером 4 і породжується стан рисунка 5.4 б). На третьому кроці виконуватиметься команда №5, яка не змінить стан машини. Оскільки секція, яка оглядається, при цьому порожня, то на четвертому кроці буде виконуватись команда з номером верхньої відсилки 4 і машина прийде в стан, зображений на рисунку 5.4 в). Тепер, на п'ятому кроці, знов виконується команда №5, проте цього разу секція, яка оглядається, вже позначена, отже, наступною буде виконуватись команда з номером, рівним значенню нижньої відсилки (команда №3). Таким чином на шостому кроці машина прийде до стану, зображеному на рисунку 5.4 г) і приступить на сьомому кроці до виконання команди №2. Оскільки команда №2 в цьому стані є нездійсненною (вона наказує стерти мітку в порожній секції) то сьомий крок буде останнім кроком роботи машини Поста, на якому відбудеться її безрезультативна зупинка.

На машині Поста можна виконувати різні дії над числами. Мова завжди йдеться про цілі додатні числа 0, 1, 2, 3, … , k .

В рамках даної алгоритмічної системи введемо поняття *лінійного масиву*. Лінійним масивом будемо називати скінченну послідовність позначених секцій стрічки, які йдуть одна за одною безперервно між двома порожніми секціями. Зрозуміло, що *довжина* або *вимірність* масиву визначається кількістю позначених секцій.

Будь-яке число n записується на стрічці машини Поста за допомогою масиву довжини $n+1$, а сам цей масив називається *машинним записом* числа n. На рисунку 5.5 наведені лінійні масиви вимірністю 5, 1 і 3, які зображують машинний запис чисел 4, 0 і 2.

Рисунок 5.5 – Приклад запису чисел на машині Поста

Розглянемо тепер декілька різних за складністю задач про додаванні одиниці на машині Поста.

I. Додавання одиниці в найпростішому випадку.

Постановка задачі: Необхідно написати програму машини Поста, яка володіє наступною властивістю. Яким би ні було число n, якщо початковий стан машини такий, що на стрічці є машинний запис числа n (а в іншому стрічка порожня) і каретка оглядає саму ліву секцію запису, то виконання програми повинно привести до результативної зупинки, після чого на стрічці (в довільному її місці) повинно бути записано число $n+1$ (а в іншому стрічка повинна бути порожньою) причому каретка може стояти де завгодно.

Позначимо через A_n сукупність всіх початкових станів машини Поста, а через E_n – сукупність всіх таких станів, в кожному з яких позначеними на стрічці є рівно $n+1$ секцій, а каретка може стояти де завгодно. Помітимо, що до E_n відносяться всі стани класу A_n і ще нескінченна множина інших станів. На рисунку 5.6 наведено декілька станів з класу A_2 (які визначають машинний запис числа 2 і відрізняються один від одного лише різним положенням масиву і жорстко зв'язаної з ним каретки відносно початку координат) і E_2.

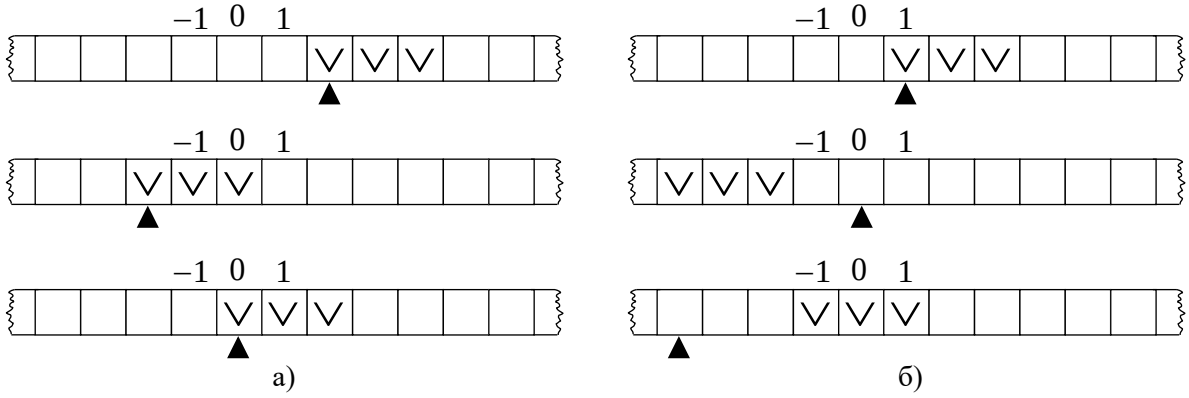


Рисунок 5.6
а) стани з класу A_2; б) стани з класу E_2

Тепер цю задачу можна сформулювати в короткій постановці: написати таку програму машини Поста, яка для будь-якого n, будучи застосована до довільного стану з класу A_n, дає результативну зупинку в будь-якому стані з класу E_{n+1}.

Розв'язком задачі I буде, наприклад, програма I_1. Цей розв'язок не єдиний, оскільки можливі інші програми, які задовольняють умовам задачі, такі, як наприклад, програми I_2 та I_3. Зазначимо, що I_1 та I_3 – дві найкоротші програми розв'язання задачі I.

Програма I_1:	Програма I_2:	Програма I_3:
1. $\Leftarrow$ 2.	1. $\Rightarrow$ 2. 4. $\Leftarrow$ 5.	1. $\Leftarrow$ 3.
2. $\vee$ 3.	2. ?(3, 3). 5. $\vee$ 6.	2. *стоп*.
3. *стоп*.	3. $\Leftarrow$ 4. 6. *стоп*.	3. $\vee$ 2.

II. Додавання одиниці в більш складних випадках.

Тепер обмежимося вимогою, щоб в початковому стані каретка оглядала одну будь-яку з секцій масиву. Позначимо через B_n множину всіх таких станів машини Поста, в кожному з яких позначеними на стрічці є рівно $n+1$ секцій, а каретка може стояти проти однієї з позначених секцій (див. рисунок 5.7).

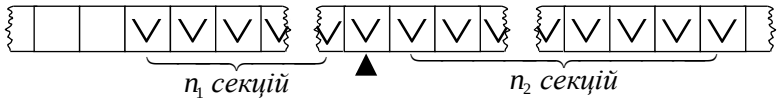


Рисунок 5.7 – Загальний вигляд станів машини Поста з класу B_n $(n_1 + n_2 = n)$

Очевидно, що клас B_n є частиною класу E_n і сам містить в собі як частину клас A_n.

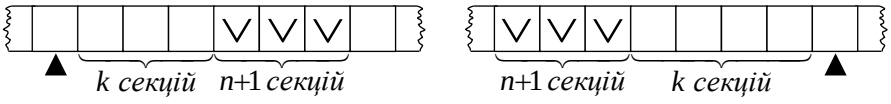
Коротке формулювання задачі II таке. Написати таку програму машини Поста, яка для будь-якого n, будучи застосована до довільного стану з класу B_n, дає результативну зупинку в якому-небудь стані з класу E_{n+1}.

Програма II_1:	Програма II_2:
1. $\Leftarrow$ 2.	1. ?(3, 2).
2. ?(3, 1).	2. $\Leftarrow$ 1.
3. $\vee$ 4.	3. $\vee$ 4.
4. *стоп*.	4. *стоп*.

Існує нескінченна множина програм, які є розв'язками задачі II. Але нас цікавлять найкоротші з них – це програми II_1 та II_2, які містять по 4 команди. Можна довести, що існує рівно 12 розв'язків задачі II, які мають довжину 4 і містять команду руху вліво (II_1, II_2, II_3, …, II_{12}). Окрім цього, програми II'_1, II'_2, II'_3, …, II'_{12}, які дістаються шляхом заміни в програмах II_1, II_2, II_3, …, II_{12} знака $\Leftarrow$ на знак $\Rightarrow$, також є розв'язками задачі II.

Розглянемо ще більш складну задачу, коли в початковому стані каретка тримає в полі зору не будь-яку з секцій початкового масиву, а будь-яку порожню секцію. При цьому будемо вважати, що заздалегідь відомо де стоїть каретка (ліворуч або праворуч від початкового масиву).

Позначимо через C_n сукупність всіх таких початкових станів машини Поста з класу E_n, в яких каретка стоїть зліва від початкового масиву, а через C'_n – сукупність всіх таких початкових станів з класу E_n, в яких каретка стоїть праворуч від масиву (див. рисунок 5.8). Таким чином, ми маємо дві задачі.



а) б)

Рисунок 5.8 – Загальний вигляд станів машини Поста: а) з класу C_n, $k \geq 0$; б) з класу C'_n, $k \geq 0$

Задача III. Написати таку програму машини Поста, яка для будь-якого n, будучи застосована до довільного стану з класу C_n, дає результативну зупинку в будь-якому стані з класу E_{n+1}.

Задача III′. Написати таку програму машини Поста, яка для будь-якого n, будучи застосована до довільного стану з класу C'_n, дає результативну зупинку в будь-якому стані з класу E_{n+1}.

Програма III_1:	Програма III'_1:
$1.\Rightarrow 2.$ $4.\vee 5.$	$1.?(2,3).$ $4.\vee 5.$
$2.?(1,3).$ $5.$ *стоп.*	$2.\Leftarrow 1.$ $5.$ *стоп.*
$3.\Leftarrow 4.$	$3.\Rightarrow 4.$

Очевидно, що кожну програму, яка є розв'язком задачі III, можна перетворити на розв'язок задачі III′, якщо усюди знак $\Leftarrow$ замінити на знак $\Rightarrow$, а знак $\Rightarrow$ замінити на знак $\Leftarrow$.

Тепер розширимо сукупність початкових станів машини Поста і тим самим ще більше ускладнимо задачу: $D_n = B_n \cup C_n$. Тобто розглянемо сукупність всіх таких станів з класу E_n, в яких секція, що оглядається кареткою, або позначена, або розташована зліва від всіх позначених.

Задача IV. Написати таку програму машини Поста, яка для будь-якого n, будучи застосована до довільного стану з класу D_n, дає результативну зупинку в будь-якому стані з класу E_{n+1}.

Програма IV_1:	
$1.?(6,2).$	$6.\Rightarrow 7.$
$2.\Leftarrow 3.$	$7.?(6,8).$
$3.?(4,2).$	$8.\Leftarrow 9.$
$4.\vee 5.$	$9.\vee 10.$
$5.$ *стоп.*	$10.$ *стоп.*

Розв'язання задачі IV можна звести до розв'язання задач II і III. Для цього достатньо перевірити, порожньою або позначеною є секція, що оглядається в початковому стані D_n. Якщо вказана секція виявилася порожньою, то необхідно виконати всі пункти програми III_1, в противному випадку необхідно діяти за алгоритмом II_1. Розв'язання задачі IV можна здійснити за програмою VI_1. Зрозуміло, що при складанні цієї програми необхідно всі команди і відсилки програми II_1 збільшити на 1, а команди і відсилки програми III_1 збільшити на $l+1$, де l – довжина програми II_1.

Таким чином, виходячи з найкоротших розв'язків задач II і III, розв'язок задачі IV не виявилось найкоротшим. Видно, що дану програму можна скоротити за допомогою так званої операції *поглинання* – об'єднання двох повторюваних команд:

$1.?(6,2).$ $6.\Rightarrow 7.$	$1.?(6,2).$ $6.\Rightarrow 7.$	$1.?(6,2).$ $6.\Rightarrow 7.$	$1.?(6,2).$ $6.\Rightarrow \not{7}.$ (1)
$2.\Leftarrow 3.$ $7.?(6,8).$	$2.\Leftarrow 3.$ $7.?(6,8).$	$2.\Leftarrow 3.$ $7.?(6,\not{8}).$ (2)	$2.\Leftarrow 3.$ ~~$7.?(6,2).$~~
$3.?(4,2).$ $8.\Leftarrow 9.$	$3.?(4,2).$ $8.\Leftarrow \not{9}.$ (4)	$3.?(4,2).$ ~~$8.\Leftarrow 4.$~~	$3.?(4,2).$
$4.\vee 5.$ $9.\vee \not{10}.$ (5)	$4.\vee 5.$ ~~$9.\vee 5.$~~	$4.\vee 5.$	$4.\vee 5.$
$5.$ *стоп.* ~~$10.$ *стоп.*~~	$5.$ *стоп.*	$5.$ *стоп.*	$5.$ *стоп.*
$\Rightarrow$	$\Rightarrow$	$\Rightarrow$	

Нарешті, остаточний результат виконаних поглинань приводить нас до програми IV_2. Абсолютно аналогічно розв'язується задача IV′ про додавання одиниці для початкових станів, в яких секція, що оглядається кареткою, або позначена, або праворуч масиву. Розв'язок задачі IV′ також можна отримати, замінивши в довільному розв'язку задачі IV знак $\Rightarrow$ на знак $\Leftarrow$, а знак $\Leftarrow$ на знак $\Rightarrow$.

Програма IV_2:	
$1.?(6,2).$	$4.\vee 5.$
$2.\Leftarrow 3.$	$5.$ *стоп.*
$3.?(4,2).$	$6.\Rightarrow 1.$

Розглянемо тепер додавання чисел на машині Поста. При цьому будемо вважати, що спочатку на стрічці нічого не записано окрім чисел $m_1, m_2, m_3, \ldots, m_k$ $(k \geq 2)$, і вимагати, щоб в кінці не було записано нічого окрім їх суми. Для простоти будемо вважати, що в початковому стані каретка стоїть проти самої лівої з позначених секцій, а на положення каретки в кінці не будемо накладати ніяких обмежень.

Задача V. Скласти програму машини Поста для додавання двох чисел, записаних на відстані однієї секції одне від одного: $m_3 = m_1 + m_2$.

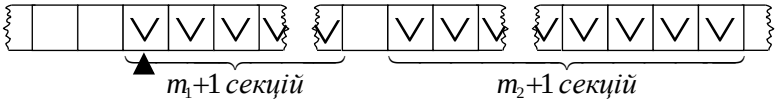


Рисунок 5.9 – Клас початкових станів машини Поста для задачі V

Простіше всього розв'язати задачу, якщо поставити мітку в порожню секцію між масивами. Тоді на стрічці виникає $(m_1+1)+(m_2+1)+1=m_1+m_2+3$ позначених секцій, тоді як їх повинне бути m_1+m_2+1. Отже, потім необхідно стерти дві зайві мітки. В програмі V_1 реалізується цей план дій.

Програма VI_1:

1. ξ 2. 6. $?(7,10)$. 11. $?(12,10)$.
2. $\Rightarrow$ 3. 7. $\Leftarrow$ 8. 12. $\Rightarrow$ 13.
3. $?(4,2)$. 8. $?(9,7)$. 13. ξ 14.
4. $\vee$ 5. 9. $\Rightarrow$ 1. 14. *стоп*.
5. $\Rightarrow$ 6. 10. $\Leftarrow$ 11.

Ми можемо написати і більш коротку програму V_2, якщо зітремо дві мітки не в кінці, а на початку прямо з лівого масиву. При цьому треба проявити обережність, щоб врахувати той випадок, коли в лівому масиві міститься всього одна мітка ($m_1 = 0$).

Програма VI_2:

1. ξ 2. 5. $\Rightarrow$ 6. 9. $?(10,12)$.
2. $\Rightarrow$ 3. 6. $?(7,5)$. 10. $\Leftarrow$ 11.
3. $?(12,4)$. 7. $\vee$ 8. 11. $?(2,10)$.
4. ξ 5. 8. $\Rightarrow$ 9. 12. *стоп*.

Задача VI. Скласти програму машини Поста для додавання двох чисел, записаних на довільній відстані одне від одного.

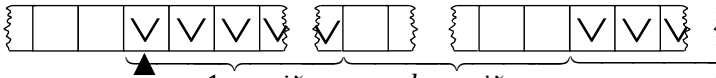


Рисунок 5.10 – Клас початкових станів машини Поста для задачі VI

Програма V_1:		Програма V_2:	
1. $\Rightarrow$ 2.	6. $\Leftarrow$ 7.	1. ξ 2.	5. $\Rightarrow$ 6.
2. $?(3,1)$.	7. ξ 8.	2. $\Rightarrow$ 3.	6. $?(7,5)$.
3. $\vee$ 4.	8. $\Leftarrow$ 9.	3. $?(8,4)$.	7. $\vee$ 8.
4. $\Rightarrow$ 5.	9. ξ 10.	4. ξ 5.	8. *стоп*.
5. $?(6,4)$.	10. *стоп*.		

Складати програму ми будемо на підставі наступної ідеї. Лівий масив пересувається вправо до тих пір, поки не зіллється з правим. Пересування масиву здійснюється шляхом перенесення самої лівої мітки масиву в найближчу порожню секцію справа. Коли масиви зливаються, то позначеними виявляються m_1+m_2+2 секції, тобто на одну секцію більше, ніж треба. Після чого, каретку необхідно перемістити в лівий кінець масиву і стерти зайву мітку. В програмі VI_1 реалізується описана ідея.

Аналогічно тому, як це було зроблено в задачі V, можна скоротити довжину програми VI_1 на дві команди, якщо стирати зайву мітку не в кінці, а на початку роботи машини Поста (див. програму VI_2).

Пора поставити питання: в яких випадках задачі на складання програми мають

розв’язки? Твердження, яке дає відповідь на поставлене питання, називається *пропозицією Поста*, що говорить: задача на складання програми, яка приводить від вхідних даних до результуючого числа, тоді і тільки тоді має розв’язок, коли існує який-небудь загальний спосіб, який дозволяє за довільними вхідними даними виписувати результуюче число. При цьому передбачається, що якщо результуючого числа не існує, то спосіб, про який йдеться (так само як і програма), не приводить ні до якого результату.

Ми вже добре знаємо, що поняття єдиного способу обчислення, загального для цілого класу можливих початкових даних є поняттям алгоритму. Це поняття тісно і природно пов'язано з машиною Поста. Як ми проілюстрували на найпростіших задачах кожна програма машини Поста, яка в застосуванні до будь-якого числа чи послідовності чисел або не дає результативної зупинки, або як результат дає число, задає деякий алгоритм.

Як видно, пропозиція Поста складається, по суті, з двох тверджень: *в першому* говориться, що з розв’язності задачі складання програми Поста виходить розв’язність задачі складання алгоритму; *в другому* говориться, що з розв’язності задачі алгоритмізації слідує розв’язність задачі програмування. Перше з цих тверджень очевидно, оскільки програма, яка служить розв’язком задачі, сама по собі утворює алгоритм. Друге твердження потребує обґрунтування і до нього, по суті діла, зводиться пропозиція Поста.
Довести твердження Поста не представляється можливим оскільки поняття алгоритму не є математично визначеним. Проте експеримент показує, що, дійсно, всякий раз, як нам задають деякий алгоритм його можна перевести у форму програми для машини Поста, яка дає той же результат.