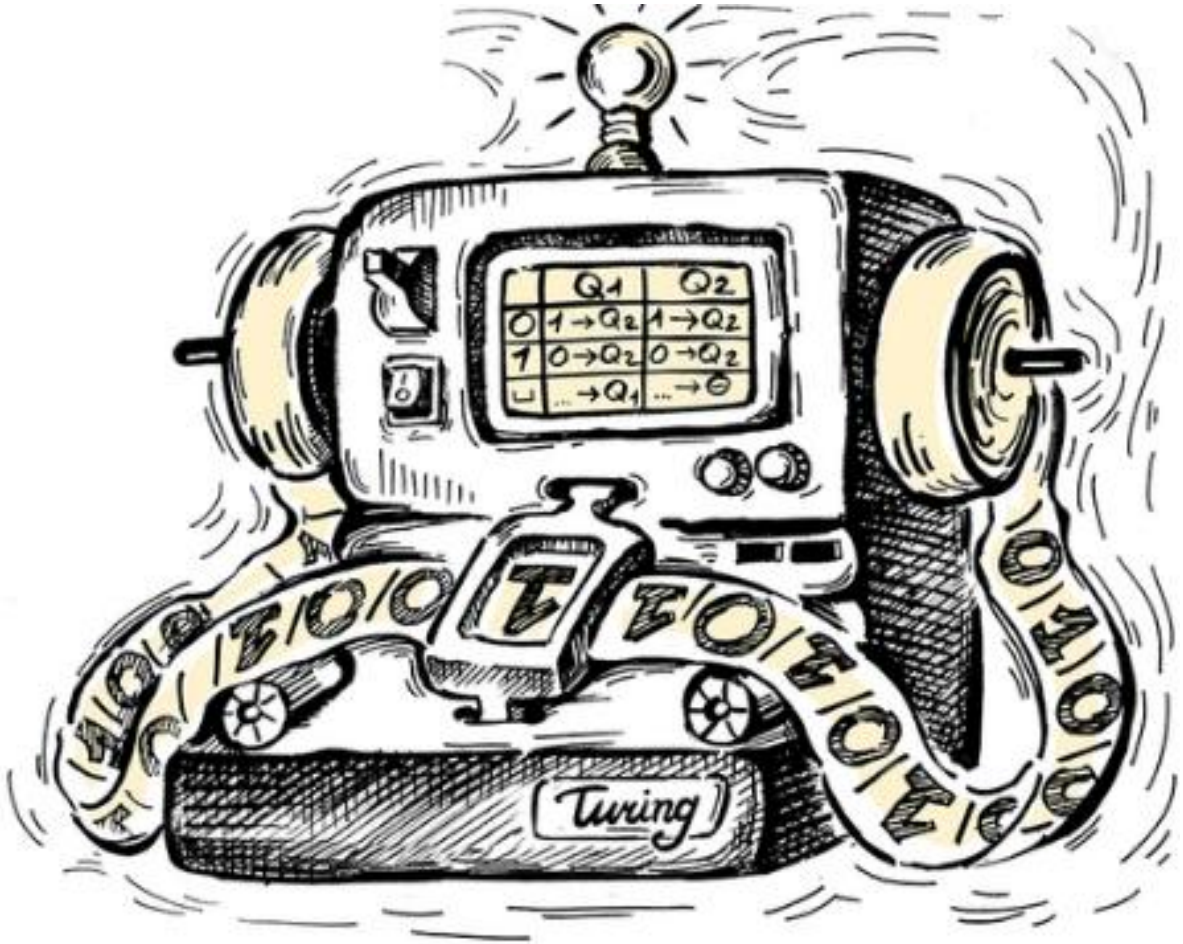
ТЕМА 3.

Найпростіша модель обчислювання. Машина Тьюринга. Автомат Маркова. Схема Поста



§1. Проблема можливості розв'язання в теорії алгоритмів

З давніх часів у математиці склалося інтуїтивне уявлення про алгоритм як формальне розпорядження, що визначає сукупність операцій і порядок їх виконання для розв'язання задач деякого типу.

До уточнення поняття «алгоритм» у математиці були поширені дві точки зору:

1. Усі проблеми є алгоритмічно розв’язними. Проте алгоритм для вирішення деяких з них не знайдено, тому що не вистачає засобів у сучасній математиці для його побудови. Прихильники цієї думки вважали, що для вирішення проблем, які називаються алгоритмічно нерозв'язними, просто не вистачає засобів сучасної математики, побудова шуканих алгоритмів — справа майбутнього.
2. Існують класи задач, для розв’язку яких взагалі не існує алгоритму, тобто деякі проблеми не можна вирішувати механічно за допомогою формальних міркувань й обчислень. Ці проблеми потребують творчого мислення. Це дуже сильне твердження, адже воно поширюється на всі майбутні часи і засоби.

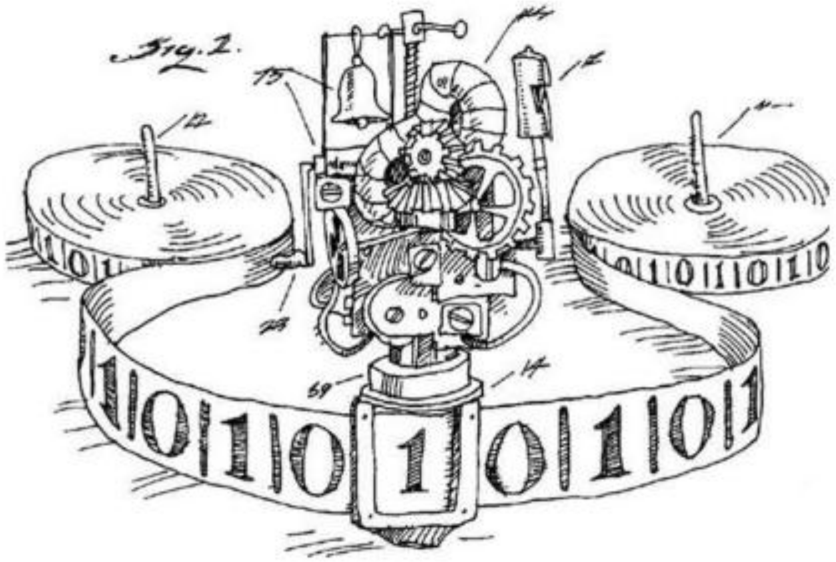
Проблема можливості розв'язання:

- Якщо задача має рішення, то відомий хоча б один алгоритм її розв'язання.
- Якщо задачу вирішити не можна, то її слід віднести до розряду алгоритмічно нерозв'язних.

Алан Тьюринг (1912-1954)

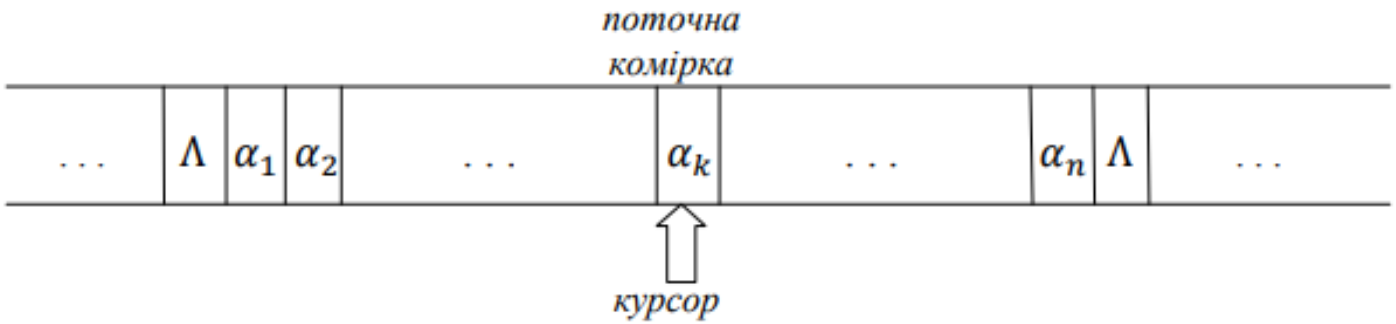
Шукав можливості доведення існування або неіснування алгоритмів вирішення різних задач.

А.Тьюрингом у 1936 р. була описана Машина Тьюринга, яка являє собою теоретичну модель обчислювальної машини.



§2. Машина Тьюринга

Машина Тьюринга - це абстрактний алгоритм, який представляється у вигляді нескінченної в обидва боки стрічки, розділеної на комірки.



Загальний вигляд машини Тьюрінга

У кожній комірці може міститися не більше одного символу з деякої скінченної множини (зовнішній алфавіт). Якщо комірка порожня, то вважають, що там міститься порожній символ, який позначається Λ (лямбда). Крім того, є курсор (вказівник, голівка, яка зчитує), який у кожний момент часу вказує на одну з комірок, яку будемо називати поточною. Машина в кожний момент часу зчитує вміст поточної комірки. Вміст решти комірок недоступний. Робота машини Тьюринга відбувається покроково. У кожен момент часу машина знаходиться в одному зі своїх (внутрішніх) станів, серед яких будемо виділяти початковий (частіше всього позначається через «q_0» та заключний (прийнято позначати через «!»).

Тут $a_o, a_1, ..., a_n$ символи зовнішнього алфавіту.

Машина Тьюринга задається:

1. **Зовнішній алфавіт** – непорожня скінченна множина символів, які можуть бути записані в комірки: $A = \{a_o, a_1, \ldots, a_n\}$.

2. **Внутрішній алфавіт** – непорожня скінченна множина внутрішніх станів машини: $Q = \{q_0, q_1, \ldots, q_n\}$ В множині Q виокремлюється початковий стан (частіше за все позначається через q_0 (та заключний стан (частіше за все позначається через !).

3. **Множина команд** – непорожня скінченна множина P={ S, L , R }. Кожна команда визначає дію машини при певному поточному стані та поточному символі.

Кожна команда має наступний вигляд: $q_m\alpha \rightarrow q_n\beta \begin{bmatrix} R \\ L \\ S \end{bmatrix}$

де q_m поточний стан, a – поточний символ.

§3. Робота машини Тьюринга

У початковий момент часу на стрічці записана скінченна послідовність символів (вхідне слово), машина знаходиться в початковому стані q_0 та, якщо не визначено інше, курсор зчитує перший символ вхідного слова.

Далі, відповідно до списку команд, курсор рухається по стрічці і слово перетворюється.

Машина Тьюринга закінчує роботу у двох випадках:

а) здійснено перехід у заключний стан (нормальне завершення роботи);
б) відсутня команда з відповідною лівою частиною (ненормальне або аварійне закінчення роботи).

Зауваження. Наприкінці роботи машини правилом гарного тону вважається повернути курсор на початок слова.

Приклад 1. До слова, яке складається із скінченної кількості паличок, дописати у кінці зірочку.

У стані q_0 ми проходимо все слово і в кінці (як тільки зустрінемо Λ) ставимо зірочку та переходимо в стан q_1. У стані q_1 ми проходимо слово справа наліво і зупиняємося на першому символі (крайня ліва паличка) і переходимо в заключний стан !.

1) $q_0| \rightarrow q_0| R$
2) $q_0\Lambda \rightarrow q_1 * L$
3) $q_1| \rightarrow q_1| L$
4) $q_1\Lambda \rightarrow !\,\Lambda\, R$

Стан	Стрічка
q_0	$\Lambda\underline{\|}\|\|\Lambda$
q_0	$\Lambda\|\underline{\|}\|\Lambda$
q_0	$\Lambda\|\|\underline{\|}\Lambda$
q_0	$\Lambda\|\|\underline{\Lambda}$
q_1	$\Lambda\|\|\underline{\|} * \Lambda$
q_1	$\Lambda\|\underline{\|}\| * \Lambda$
q_1	$\Lambda\underline{\|}\|\| * \Lambda$
q_1	$\underline{\Lambda}\|\|\| * \Lambda$
!	$\Lambda\underline{\|}\|\| * \Lambda$

§4. Способи задання машини Тьюринга

Машина Тьюринга може задаватися:

1. списком команд (приклад 1);
2. табличним способом;
3. графічним способом.

Табличний спосіб задання машини Тьюрінга

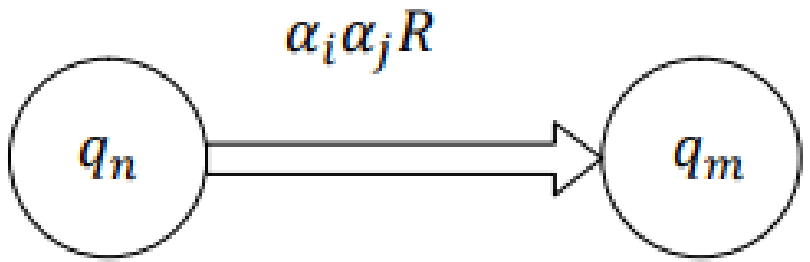
	q_0	q_1	…	q_n	…	q_N	!
α_1			…		…		
α_2			…		…		
…	…	…	…	…	…	…	
α_i			…	$q_m\alpha_j\ R$	…		
…	…	…	…	…	…	…	
α_M			…		…		
Λ			…		…		

тут $Q = \{q_0, q_1, \dots, q_N, !\}$,

$A = \{\alpha_1, \alpha_2, \dots, \alpha_M\}$.

Команда $q_n\alpha_i \to q_m\alpha_j\ R$ відображена в табл. У відповідну клітинку записується права частина команди.

Графічний спосіб: Машина Тьюринга зображується у вигляді орієнтованого міченого мультографа, де множина вершин співпадає з множиною станів, а кожна команда зображується у вигляді ребра. Наприклад, команда $q_n\alpha_i \rightarrow q_m\alpha_j\ R$ зображується таким чином:



Зображення команди машини Тьюрінга

Приклад 2. Обчислити x+1 у двійковій системі числення.

У стані q_0 ми проходимо число до кінця, переходимо в стан q_1 і стаємо на останню цифру. Якщо ця цифра – «1», то замість неї пишемо «0» і стан не змінюємо, бо з усіма наступними одиницями виконуємо цю саму дію. Але якщо зустрічаємо «0», то замість нього пишемо «1» і змінюємо стан на q_2.У стані q_2 ми проходимо число справа наліво і зупиняємося на першій цифрі отриманого числа. Окремий випадок, коли ми зустрінемо Λ у стані q_1. Цей випадок означає, що ми маємо справу з числом вигляду 11 … 1. Тут всі «1» замінилися на «0», і тоді залишається замість Λ поставити «1» і перейти в заключний стан.

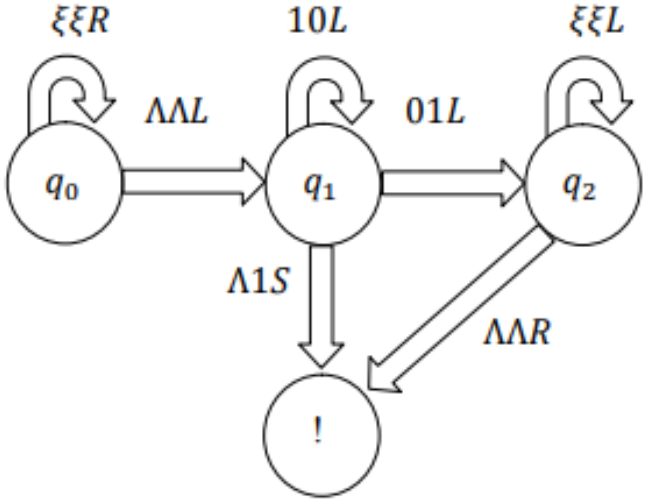
1) $q_0\xi \to q_0\xi\, R$
2) $q_0\Lambda \to q_1\Lambda\, L$
3) $q_1 0 \to q_2 1\, L$
4) $q_1 1 \to q_1 0\, L$
5) $q_1\Lambda \to !\, 1\, S$
6) $q_2\xi \to\ q_2\xi\, L$
7) $q_2\Lambda \to !\, \Lambda\, R$

$\xi \in \{0; 1\}$ використовується для скорочення запису. Тобто команда $q_0\xi \to q_0\xi\, R$ означає дві команди: $q_0 0 \to q_0 0\, R$ та $q_0 1 \to q_0 1\, R$.

Табличний спосіб

	q_0	q_1	q_2	!
0	$q_0 0R$	$q_2 1L$	$q_2 0L$	
1	$q_0 1R$	$q_1 0L$	$q_2 1L$	
Λ	$q_1 \Lambda L$	$1!S$	$\Lambda! R$	

Графічний спосіб



§5. Нормальні алгоритми Маркова.
Опис нормального алгоритму Маркова

Нормальний алгоритм Маркова (НАМ) задається наступним чином.

Нехай $\mathcal{A}$ – алфавіт (непорожня скінченна множина символів) $\mathcal{A} \neq \emptyset$, $|\mathcal{A}| < \infty$. НАМ записується у вигляді лінійно впорядкованої множини підстановок P ($P \neq \emptyset$, $|P| < \infty$).

$$P = \{\alpha_i \longrightarrow \beta_i \mid \alpha_i, \beta_i \in \mathcal{A}^* \; i = \overline{1,n}\}$$

Тут $\mathcal{A}^*$ – множина всіх слів над алфавітом $\mathcal{A}$. Слово – скінченна послідовність символів з даного алфавіту. $\mathcal{A}^*$ містить також порожнє слово (не містить жодного символу), яке будемо позначати через ε. У множині P обов'язкова наявність хоча б однієї заключної підстановки, яку будемо позначати через $\alpha_k \dot{\longrightarrow} \beta_k$.

Окремо взята підстановка $\alpha_i \longrightarrow \beta_i$ застосовується у випадку, коли поточне слово містить α_i як підслово. Застосування підстановки полягає в тому, що перше входження α_i замінюється на β_i. Порожнє підслово ε міститься між будь-якими літерами даного слова і перше його входження міститься перед першою літерою слова.

Робота нормального алгоритму Маркова

На кожному кроці до поточного слова використовується завжди перша підстановка, яку можна застосувати.

Алгоритм закінчує роботу у двох випадках:

а) використали одну із заключних підстановок (нормальне завершення роботи);

б) жодну з підстановок не можна застосувати до поточного слова (ненормальне, або аварійне, завершення роботи).

Приклад 3. Анулювати слово в алфавіті $\{a,b\}$.

Спочатку застосовується підстановка 1). Видаляються всі літери a. Потім підстановка 2). Видаляються всі літери b. І після цього підстановка 3) (заключна).

1) $a \longrightarrow \varepsilon$

2) $b \longrightarrow \varepsilon$

3) $\varepsilon \stackrel{\cdot}{\longrightarrow} \varepsilon$

Розберемо цей приклад покроково для слова $aaba$.

№ кроку	Підстановка	Слово
0	–	$\underline{a}aba$
1	1)	$\underline{a}ba$
2	1)	$b\underline{a}$
3	1)	$\underline{b}$
4	2)	$\underline{\varepsilon}$
5	3)	ε

Приклад 4. Інвертувати слово в алфавіті $\{a,b\}$. Замість a пишеться b, а замість b пишеться a.

У вхідному слові зірочки немає, тому використовується підстановка 4). Ставимо зірочку на початку слова. Потім, використовуючи підстановки 1) та 2), рухаємо зірочку вправо і замінюємо по дорозі кожну літеру a на b і літеру b на a. Якщо після зірочки немає жодної літери, застосовуємо 3). Це заключна підстановка, тому закінчуємо роботу алгоритму.

1) $* a \longrightarrow b *$

2) $* b \longrightarrow a *$

3) $* \stackrel{\cdot}{\longrightarrow} \varepsilon$

4) $\varepsilon \longrightarrow *$

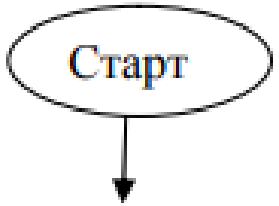
Розберемо цей приклад покроково для слова *abaa*

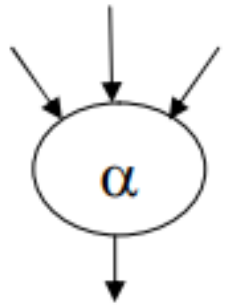
№ кроку	Підстановка	Слово
0	–	$\underline{\varepsilon}abaa$
1	4)	$\underline{*}abaa$
2	1)	$b\underline{*}baa$
3	2)	$ba\underline{*}aa$
4	1)	$bab\underline{*}a$
5	1)	$babb\underline{*}$
6	3)	$babb$

§6. Блок-схема Поста. Опис блок-схеми Поста

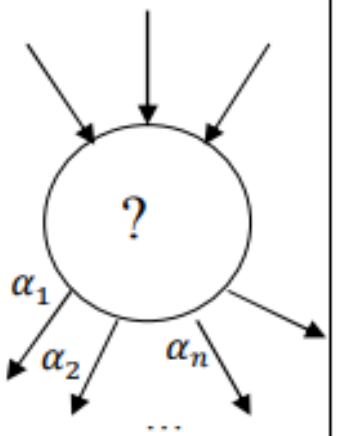
Блок-схема Поста задається наступним чином. Нехай $\mathcal{A}$ – алфавіт ($\mathcal{A} \neq \emptyset$, $|\mathcal{A}| < \infty$). БСП – алгоритмічна машина, яка задається у вигляді міченого орієнтованого мультиграфа, який може містити 4 типи вершин. На кожному кроці керування знаходиться на деякій вершині .

Таблиця вершин для БСП

№	Назва	Позначення	Дія	Примітка
1	Стартова		Жодна дія не виконується, означає початок роботи алгоритму.	Одна вершина для всього алгоритму, вхідних ребер немає, вихідне – одне.

2	Породжуюча (генеруюча) вершина		Записуємо символ $\alpha \in \mathcal{A}$ у кінець поточного слова.	Вхідних ребер може бути багато, але мінімум одне, вихідне ребро – одне.

4	Заключна вершина		Жодна дія не виконується, означає закінчення роботи алгоритму.	Вхідних ребер – не менше одного, вихідних – немає. Можлива наявність кількох заключних вершин.

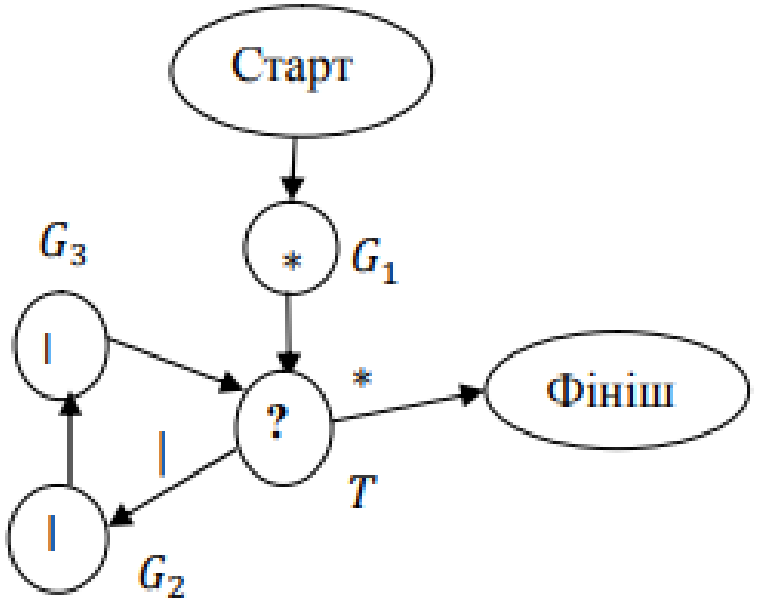
3	Тестова вершина		Видаляємо перший символ. Керування передаємо по ребру, що відповідає цьому символу.	$? \notin \mathcal{A}$, $\mathcal{A} = \{\alpha_1 \dots \alpha_n\}$. Вхідних ребер не менше одного, вихідних не більше $n+1$, де $n = \lvert A \rvert$. Ще одне ребро для випадку порожнього слова. Якщо деякий символ не зустрічається при керуванні даної вершини, то відповідне ребро можна не малювати.

§6. Робота блок-схеми Поста

На вхід задається вхідне слово скінченної довжини із заданого алфавіту $\mathcal{A}$.БСП працює покроково. На кожному кроці рівно одна вершина є поточною. В початковий момент часу поточною є стартова вершина. При переході до наступного кроку над словом проводиться деяка дія відповідно до поточної вершини. Далі керування передається наступній вершині (відповідно вихідного ребра), наявність петель допускається. Алгоритм закінчує роботу, коли поточною є одна з заключних вершин.

Приклад 5. Збільшити кількість паличок у два рази.

Ставимо у кінці слова зірочку. А далі видаляємо першу паличку у поточному слові і в кінець записуємо дві палички. Алгоритм закінчує роботу, коли при керуванні тестової вершини зустрінеться зірочка.



Розберемо цей приклад покроково для слова |||.

Протокол роботи БСП

№ кроку	Поточна вершина	Поточне слово
0	*Старт*	|||
1	G_1	|||*
2	T	||*
3	G_2	||*|
4	G_3	||*||
5	T	|*||
6	G_2	|*|||
7	G_3	|*||||
8	T	*||||
9	G_2	*|||||
10	G_3	*||||||
11	T	||||||
12	*Фініш*	||||||