

Machines de Turing ? Automates finis ? à pile ?

Visite guidée du bestiaire informatique

Nicolas Schabanel

Chargé de recherches CNRS

Laboratoire de l'informatique du parallélisme

École normale supérieure de Lyon

L'informatique est née sur le papier 1931-1936

1900 Hilbert : *Peut-on prouver tous les théorèmes mathématiques vrais ?*

1931 : Théorème d'incomplétude de Gödel

1932-1936 : λ -calcul de Church & Kleene

1936 : **Machine de Turing**

1936 : **Thèse de Church & Turing**

L'informatique est née sur le papier 1931-1936

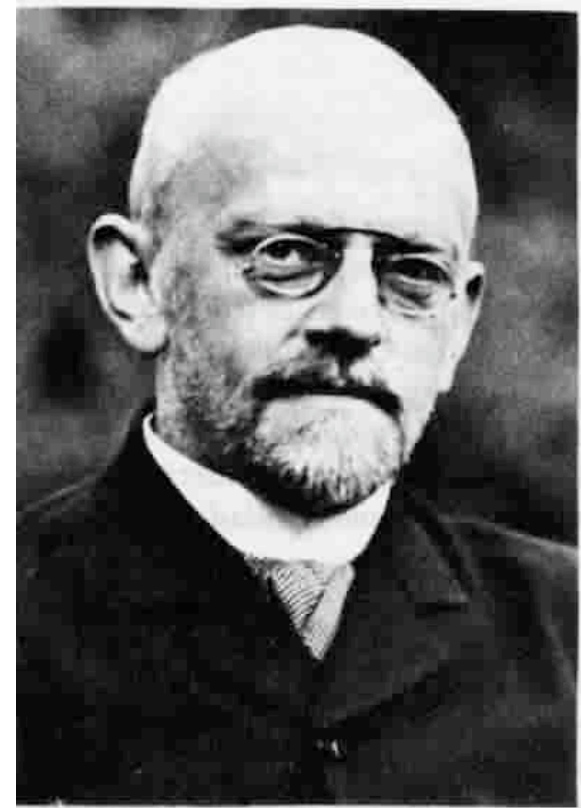
1900 Hilbert : *Peut-on prouver tous les théorèmes mathématiques vrais ?*

1931 : Théorème d'incomplétude de Gödel

1932-1936 : λ -calcul de Church & Kleene

1936 : **Machine de Turing**

1936 : **Thèse de Church & Turing**



Hilbert

L'informatique est née sur le papier 1931-1936

1900 Hilbert : *Peut-on prouver tous les théorèmes mathématiques vrais ?*

1931 : Théorème d'incomplétude de Gödel

1932-1936 : λ -calcul de Church & Kleene

1936 : Machine de Turing

1936 : Thèse de Church & Turing

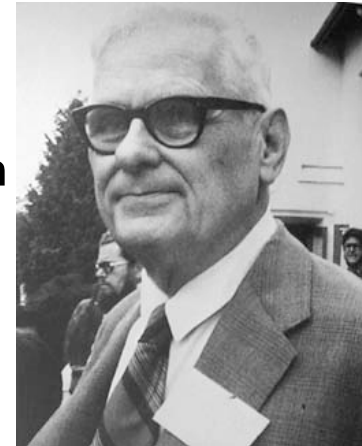


Gödel

L'informatique est née sur le papier 1931-1936

1900 Hilbert : *Peut-on prouver tous les théorèmes mathématiques vrais ?*

Church



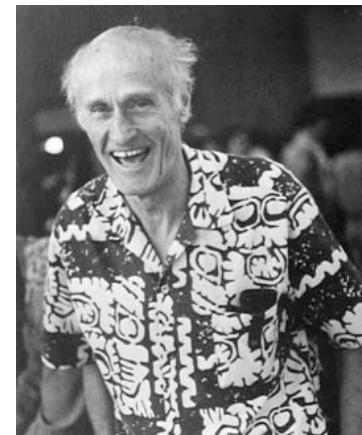
1931 : Théorème d'incomplétude de Gödel

1932-1936 : λ -calcul de Church & Kleene

1936 : Machine de Turing

1936 : Thèse de Church & Turing

Kleene



L'informatique est née sur le papier 1931-1936

1900 Hilbert : *Peut-on prouver tous les théorèmes mathématiques vrais ?*

1931 : Théorème d'incomplétude de Gödel

1932-1936 : λ -calcul de Church & Kleene

1936 : **Machine de Turing**

1936 : **Thèse de Church & Turing**



Turing

Les premiers ordinateurs (1944-)

1943 : Bomb, décryptage des codes allemands

1944 : ENIAC, premier calculateur

1945 : Architecture de von Neumann, Mauchly & Eckert

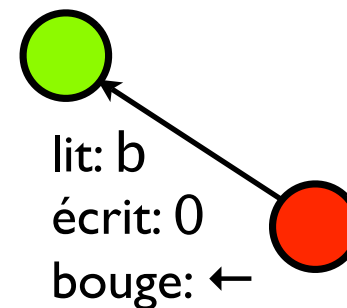
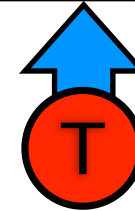
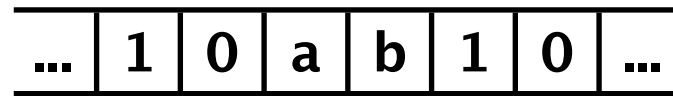
1947 : EDSAC/IAS/EDVAC, premiers ordinateurs électroniques (à lampes)

1954 : Premier langage de haut niveau, **FORTRAN**

Un ordinateur ressemble à ça pour ses créateurs

Une machine de Turing

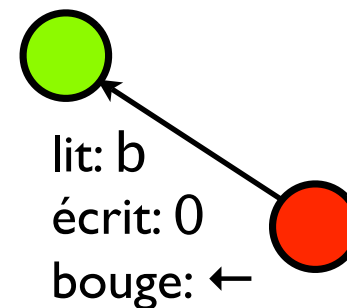
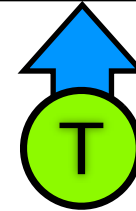
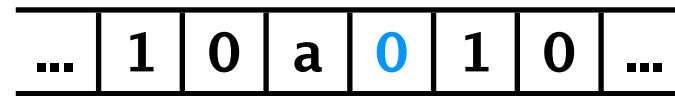
- un **ruban infini**
- une **tête T** de lecture/écriture,
- un **automate fini A** avec un **nombre fini d'états** qui change d'état et déplace la tête en fonction de la lettre lue et de l'état courant



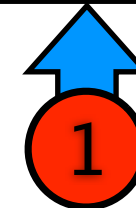
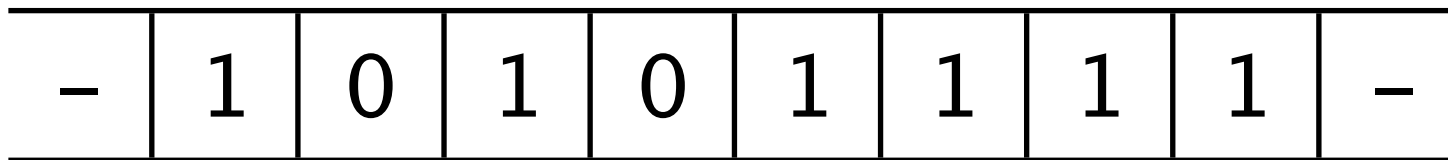
Un ordinateur ressemble à ça pour ses créateurs

Une machine de Turing

- un **ruban infini**
- une **tête T** de lecture/écriture,
- un **automate fini A** qui change d'état et déplace la tête en fonction de la lettre lue et de l'état courant



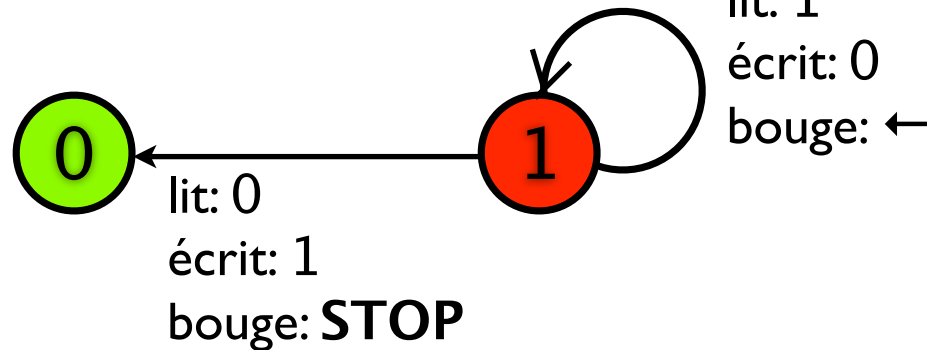
Ajouter 1 à un compteur avec une machine de Turing



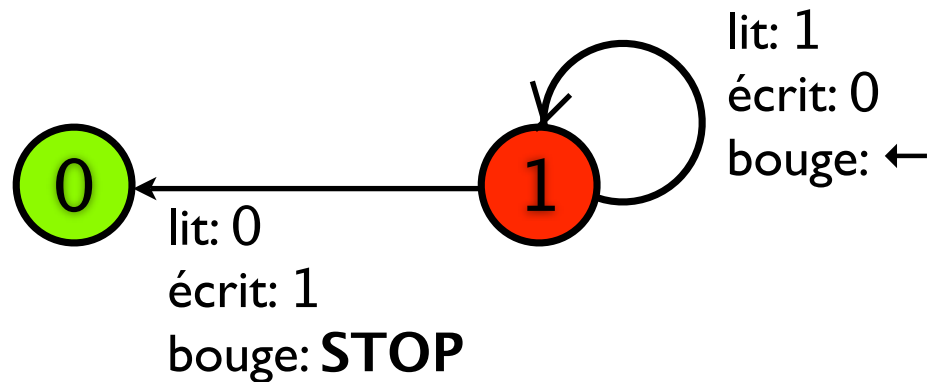
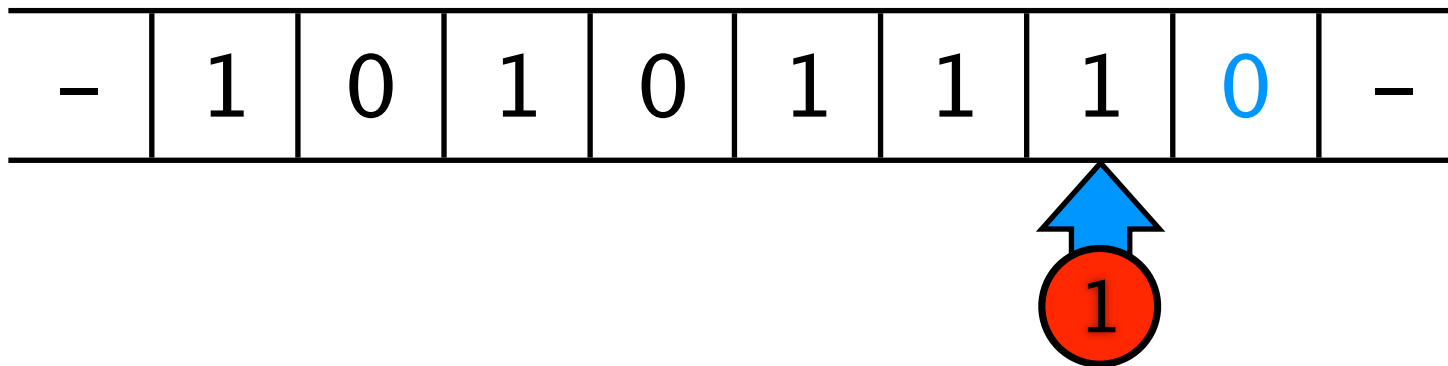
2 états:

la retenue vaut 1

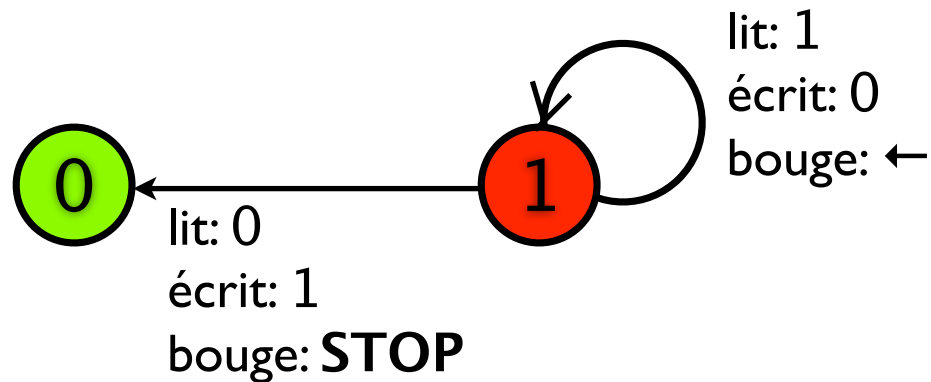
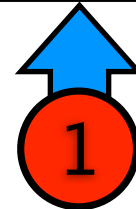
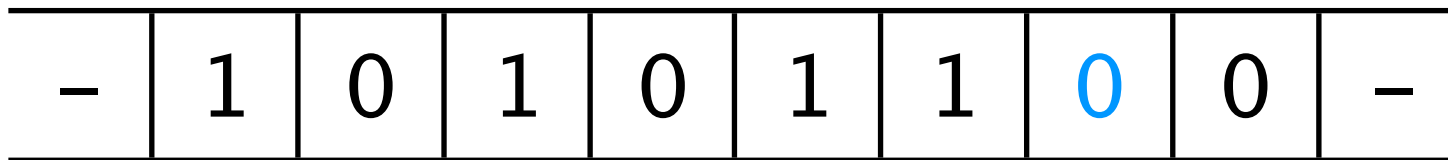
la retenue vaut 0



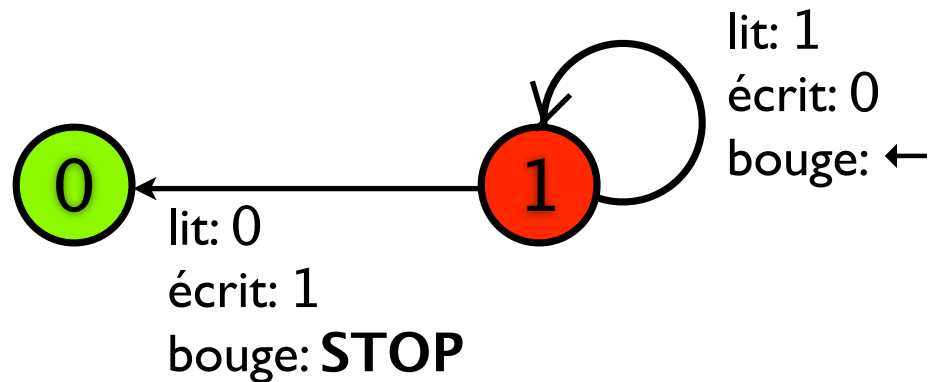
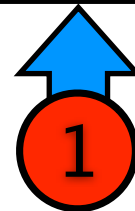
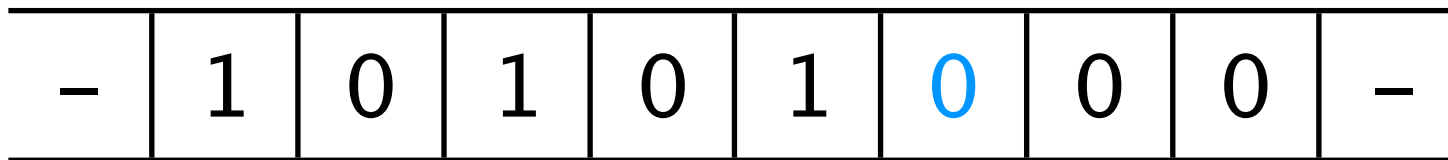
Ajouter 1 à un compteur avec une machine de Turing



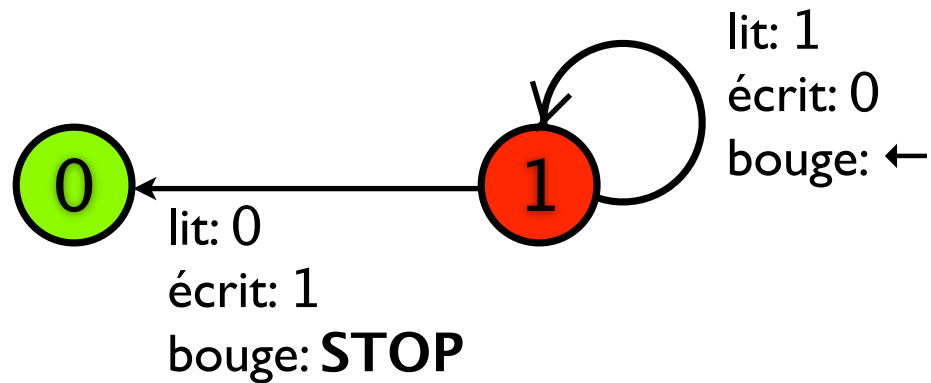
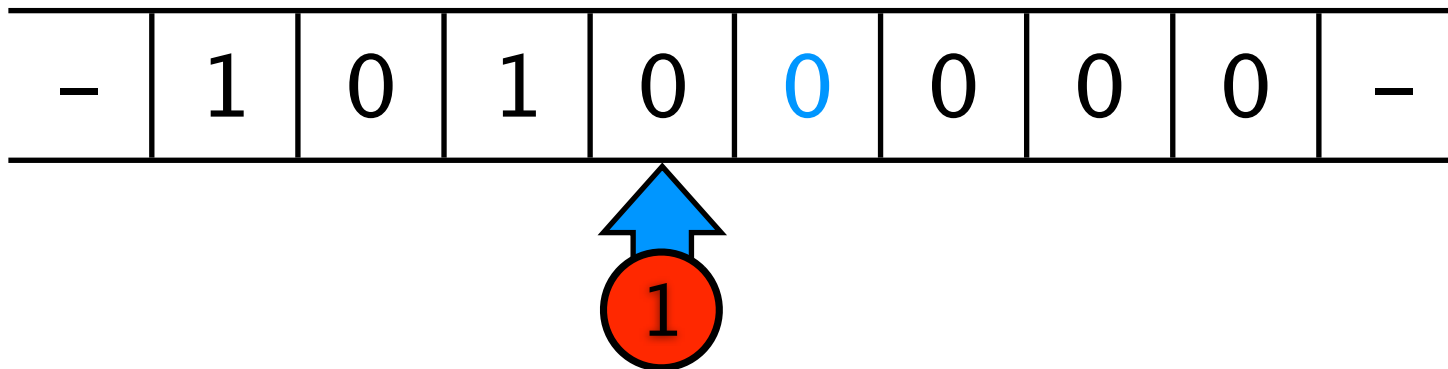
Ajouter 1 à un compteur avec une machine de Turing



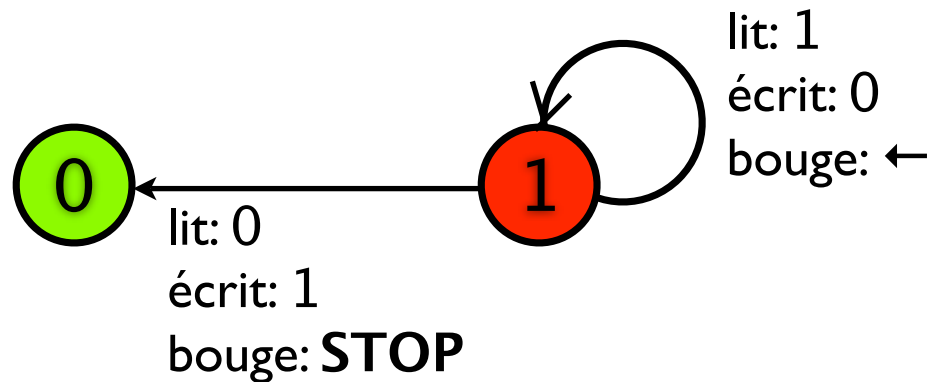
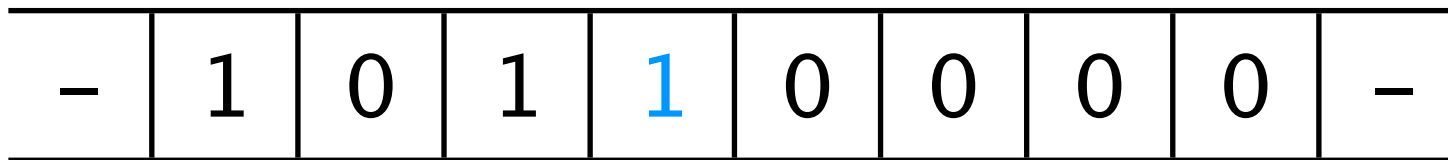
Ajouter 1 à un compteur avec une machine de Turing



Ajouter 1 à un compteur avec une machine de Turing



Ajouter 1 à un compteur avec une machine de Turing



Thèse de Church-Turing (1936)

***« Tout ce qui est calculable est
calculable par une machine de Turing »***

Mais un ordinateur a une mémoire finie !?!

Un ruban de longueur finie = pas de ruban

Les états possibles sont :

(état de la tête, ruban, position de la tête)

Il y en a qu'un nombre *fini* !

- ➔ On peut tout coder dans les états du graphe de l'automate !

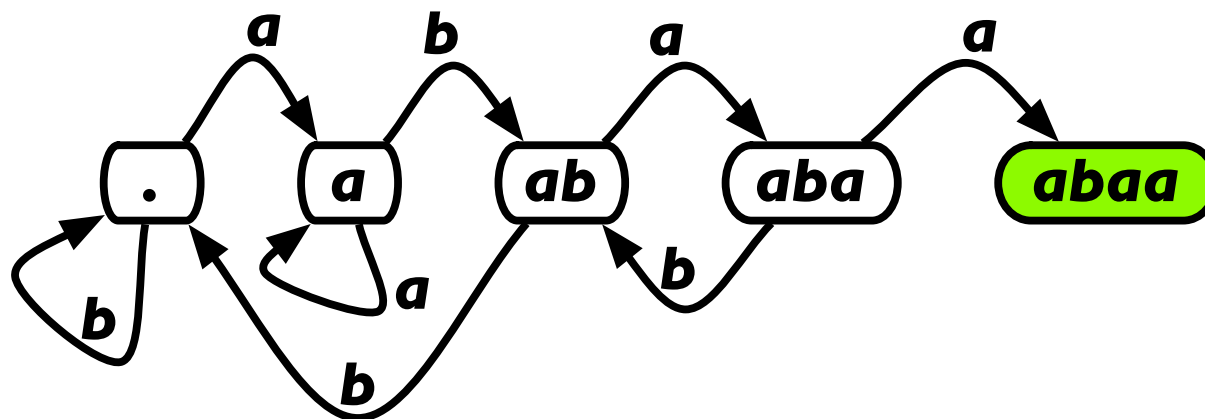
Automates finis

- Servent principalement à reconnaître et rechercher des mots dans un texte
- **Exemple : *abaa***

↳ *pas besoin d'une mémoire infinie pour cela*

Automates finis

- Servent principalement à reconnaître et rechercher des mots dans un texte
- **Exemple : *abaa***



↳ *pas besoin d'une mémoire infinie pour cela*

Automates finis

- **Exemples de familles de mots reconnues**
 - les mots qui ont un nombre pair de ***a***
 - ****.txt*** les mots qui se terminent par ***.txt***
 - ***(aba)ⁿ*** les mots constitués de répétitions du motif ***aba***

Existe-t-il des mots qui ne soient pas reconnaissables ?

Lemme de la pompe

Les devises Shadok



IL VAUT MIEUX POMPER MÊME S'IL NE SE PASSE
RIEN QUE RISQUER QU'IL SE PASSE QUELQUE CHOSE
DE PIRE EN NE POMPANT PAS.

Existe-t-il des mots qui ne soient pas reconnaissables ?

Lemme de la pompe

Chaque mot lu correspond un chemin dans le graphe de l'automate

Un mot est reconnu ssi ce chemin atteint un état d'acceptation

Si le mot a plus de lettres que le nombre d'états, alors ce chemin fait une **boucle** !

On peut répéter la boucle un nombre arbitraire de fois, i.e. pomper sur ce cycle créer de nouveaux mots reconnus !

Existe-t-il des mots qui ne soient pas reconnaissables ?

les mots correctement parenthésés ne sont pas reconnaissables par un automate fini !

Par l'absurde, soit q le nombre d'états de l'automate, alors la lecture de

$$\underbrace{((\dots(}_{q+1 \text{ fois}} \underbrace{)\dots))}_{q+1 \text{ fois}}$$

donne une boucle de longueur $k \geq 1$ sur les q premières (et donc en pompant sur cette boucle, l'automate acceptera également le mot suivant qui n'est pas correct !

$$\underbrace{((\dots(}_{q+1+k \text{ fois}} \underbrace{)\dots))}_{q+1 \text{ fois}}$$

Il est important qu'un ordinateur ait une mémoire potentiellement infinie

- Une approche raisonnable : la pile



Automates à pile (sans «s»)



Automates à pile (sans «s»)



Automates à pile (sans «s»)

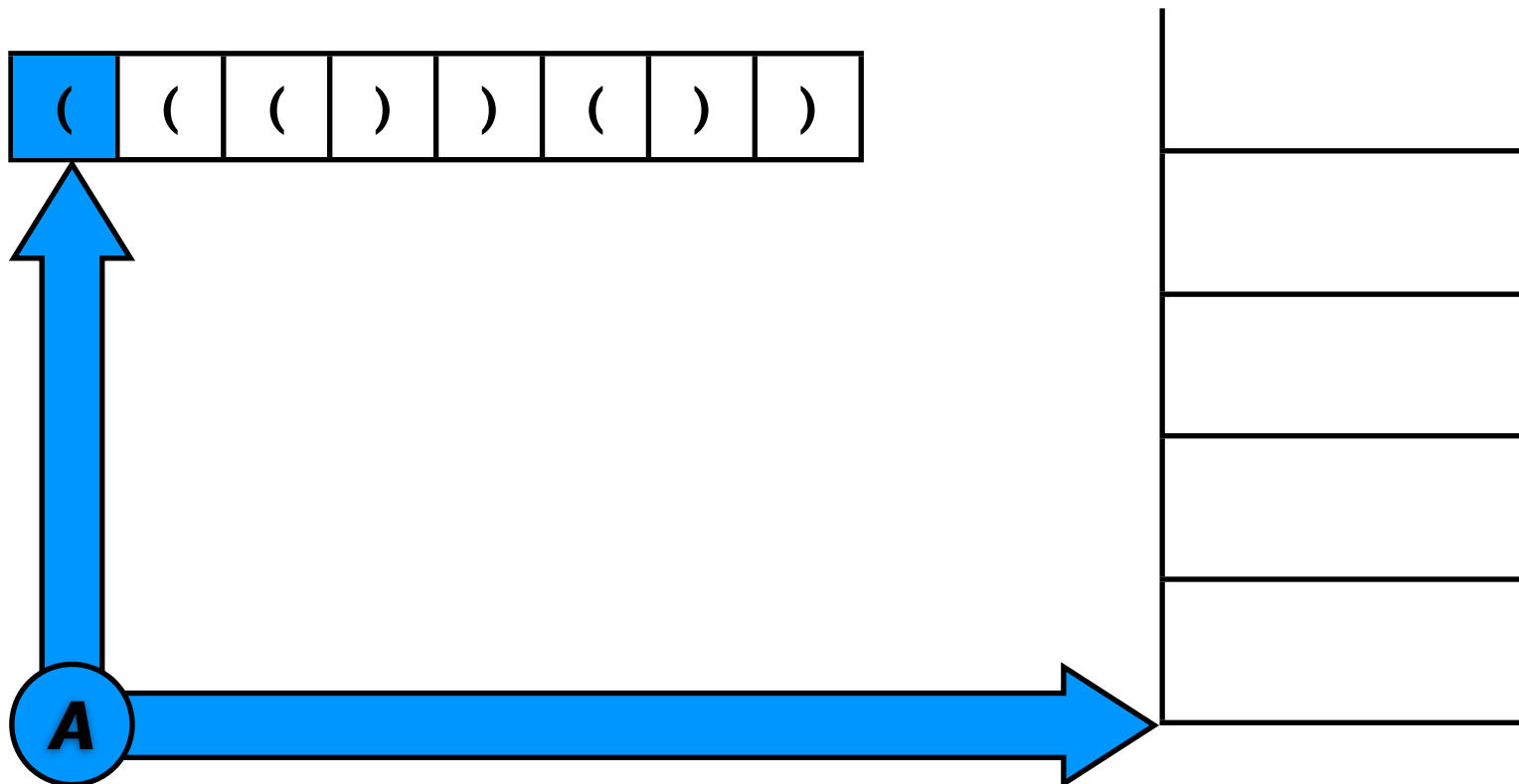
Un automate fini qui peut **écrire** et **lire** des données dans une **pile**

la pile = mémoire potentiellement infinie

limitation : ne peut lire et écrire qu'en haut de la pile

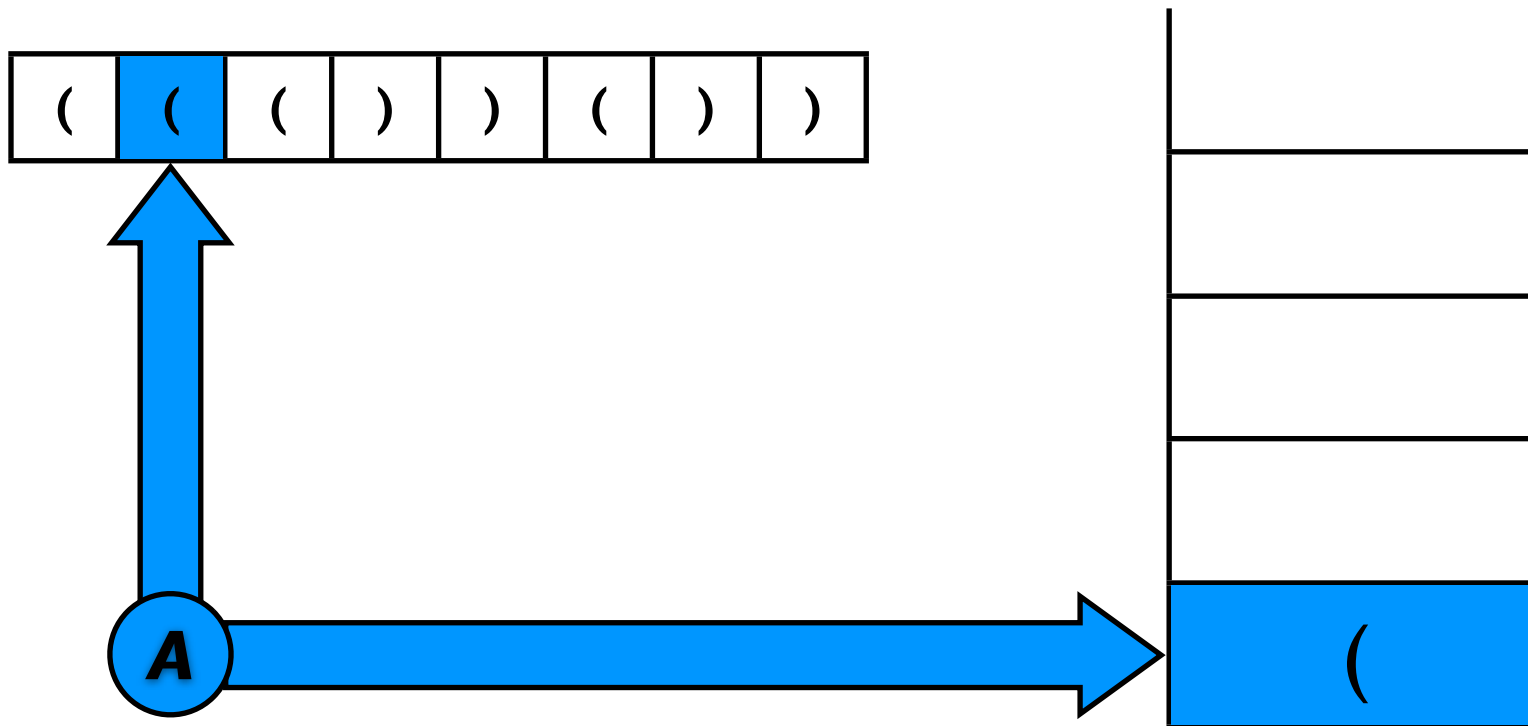
Automates à pile (sans «s»)

Reconnait les mots correctement parenthésés



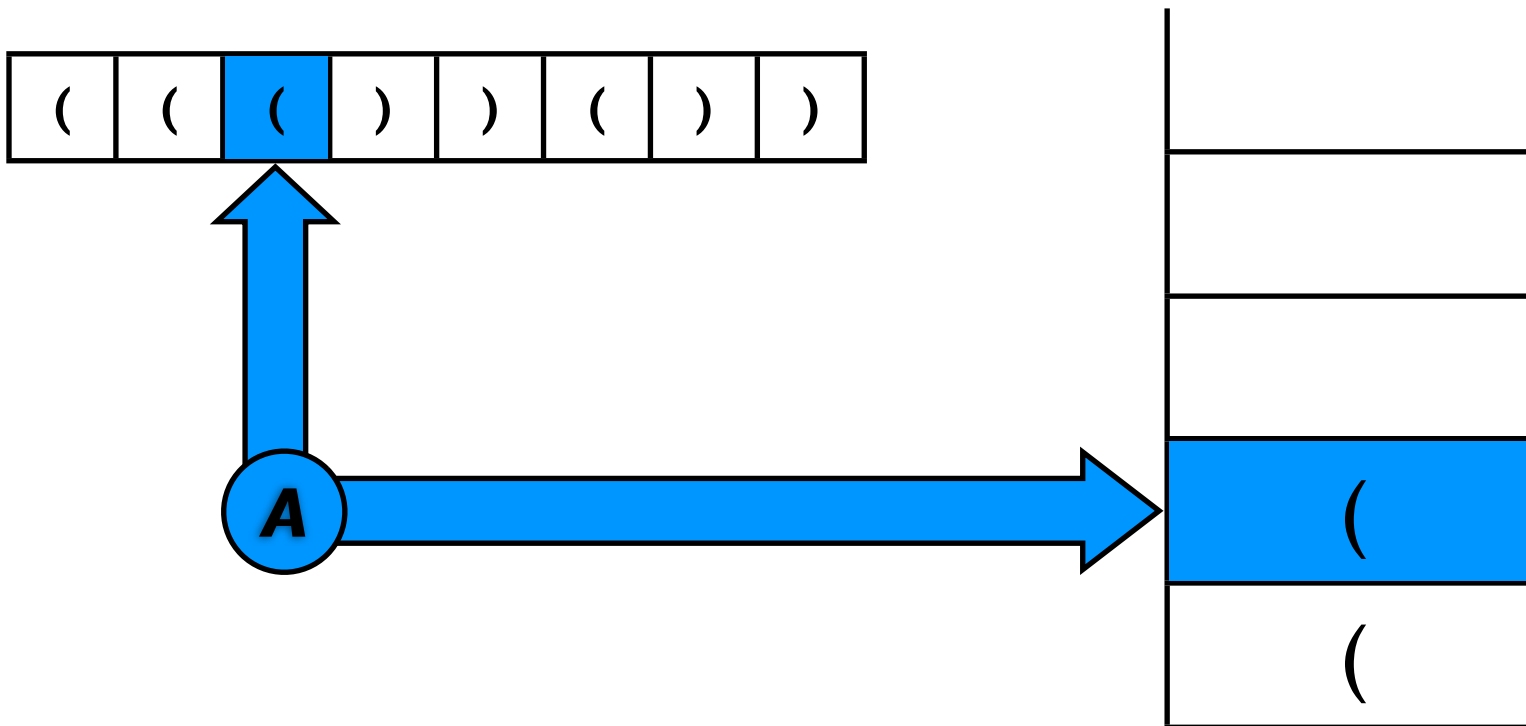
Automates à pile (sans «s»)

Reconnaît les mots correctement parenthésés



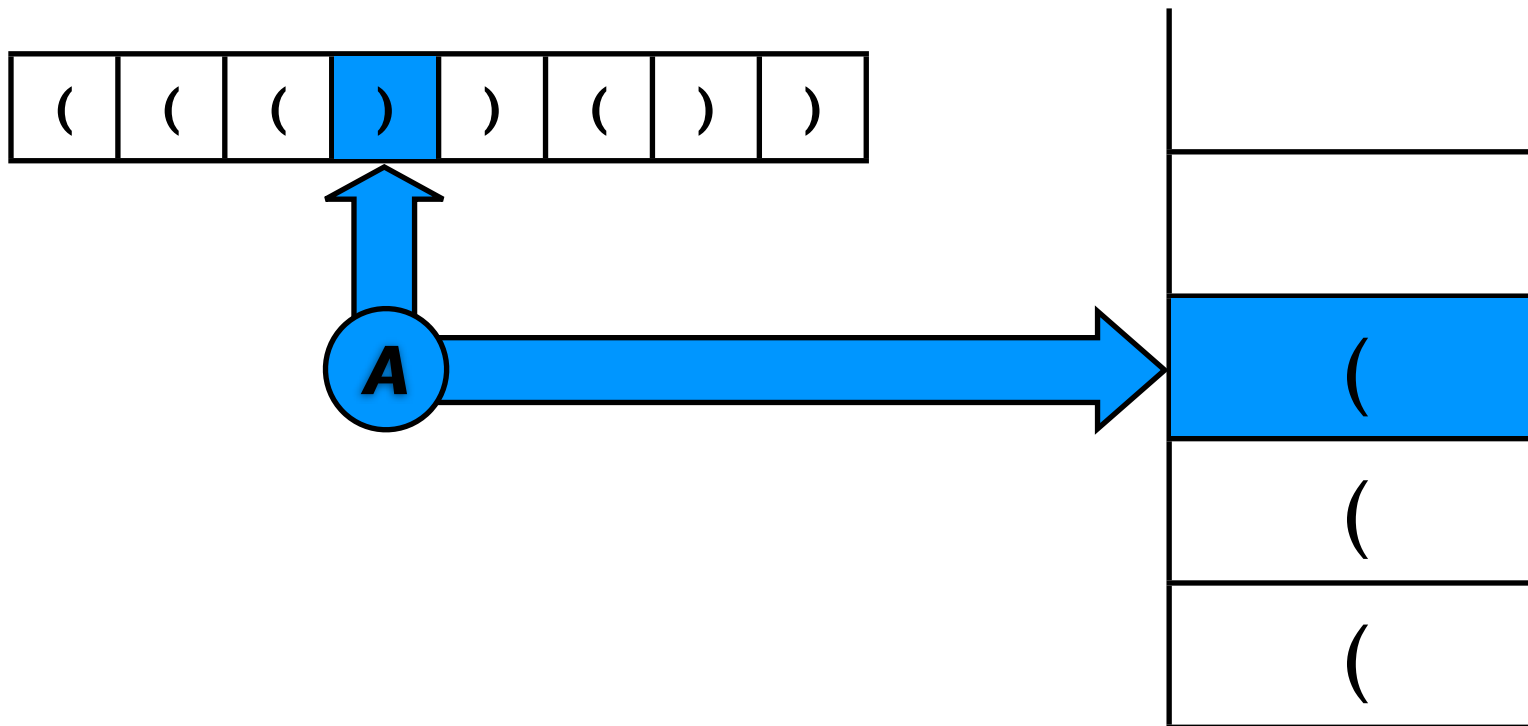
Automates à pile (sans «s»)

Reconnaît les mots correctement parenthésés



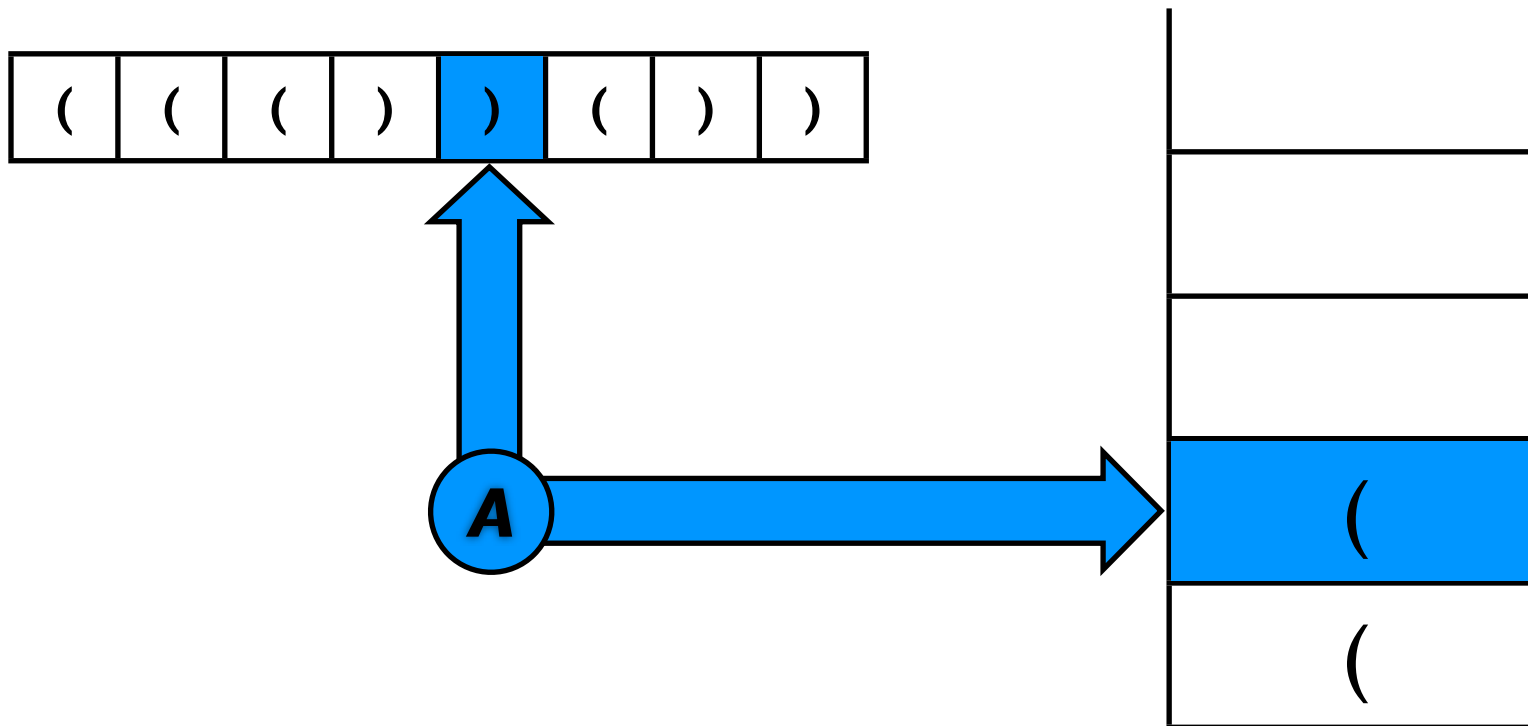
Automates à pile (sans «s»)

Reconnaît les mots correctement parenthésés



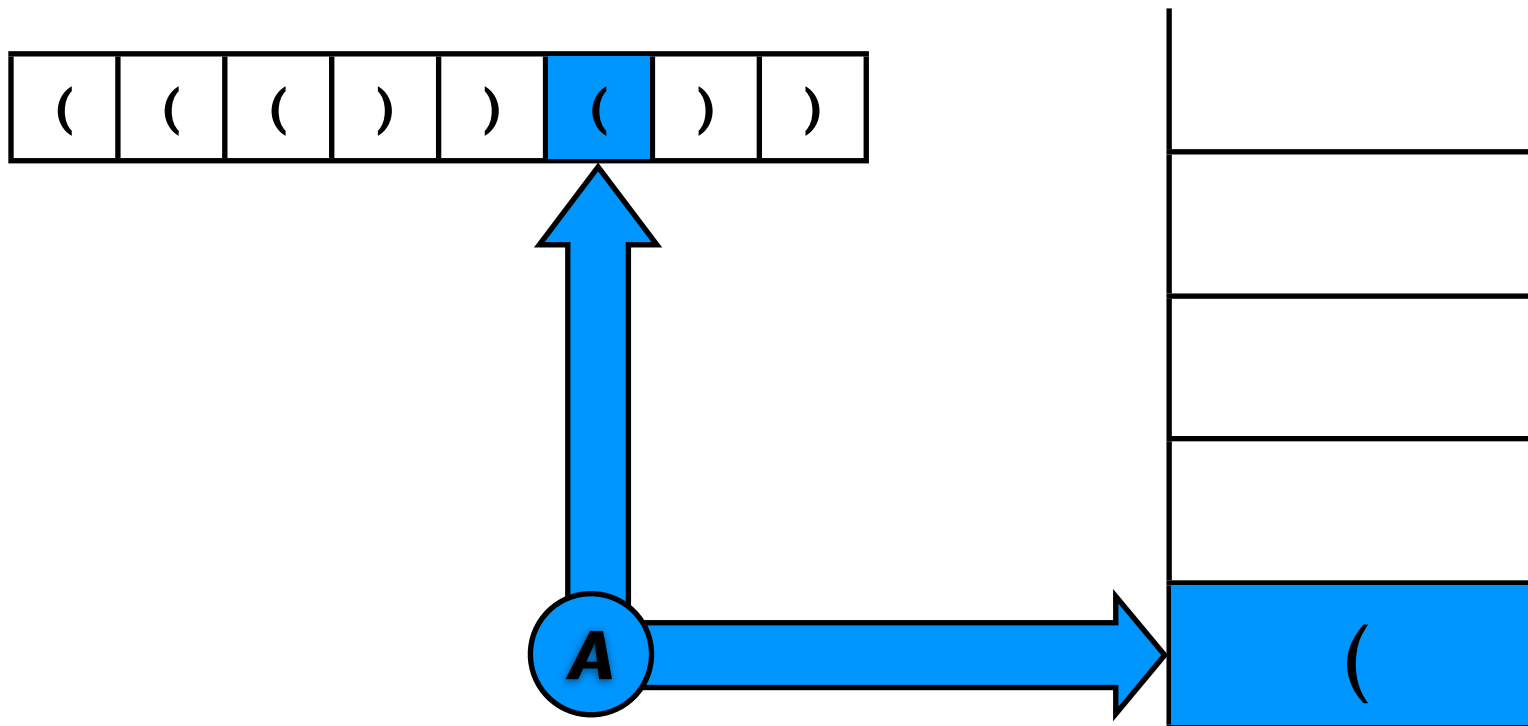
Automates à pile (sans «s»)

Reconnaît les mots correctement parenthésés



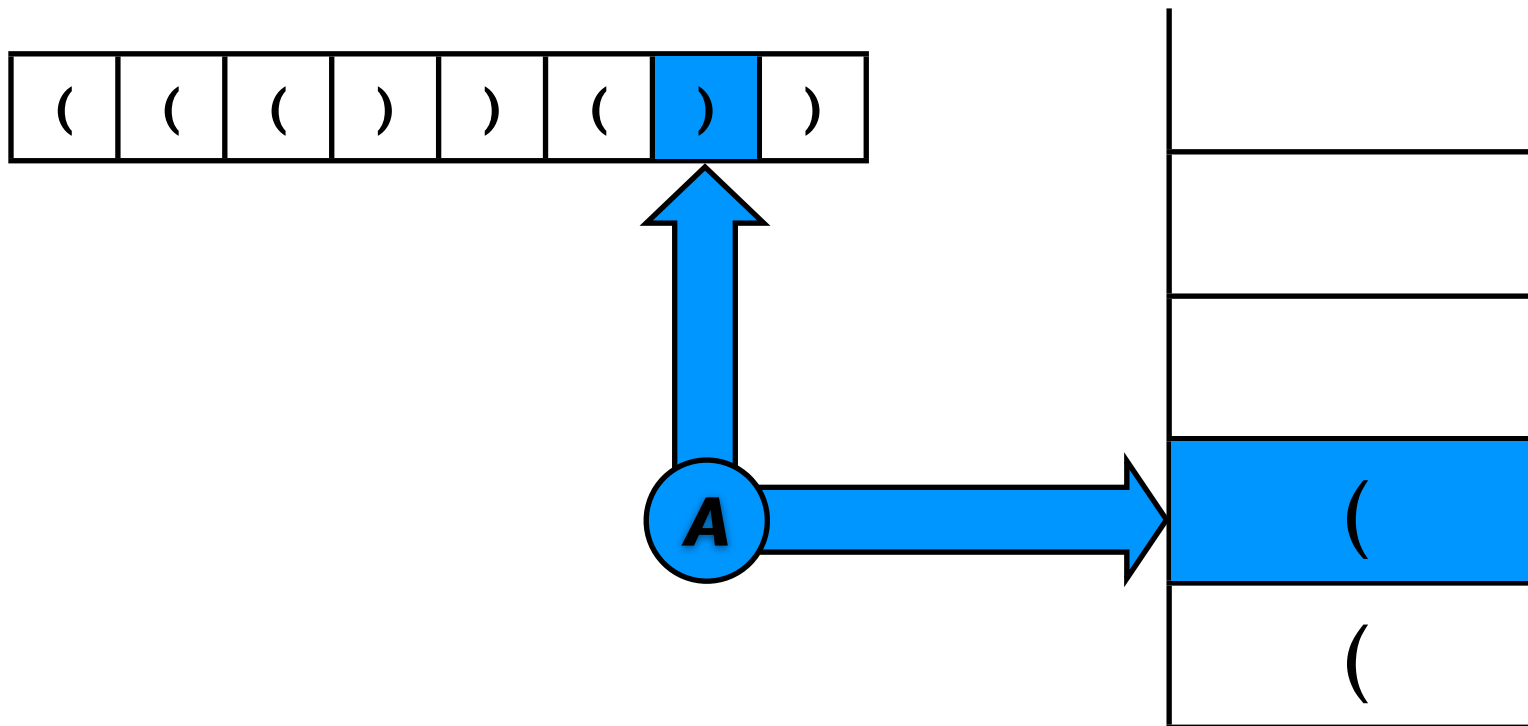
Automates à pile (sans «s»)

Reconnaît les mots correctement parenthésés



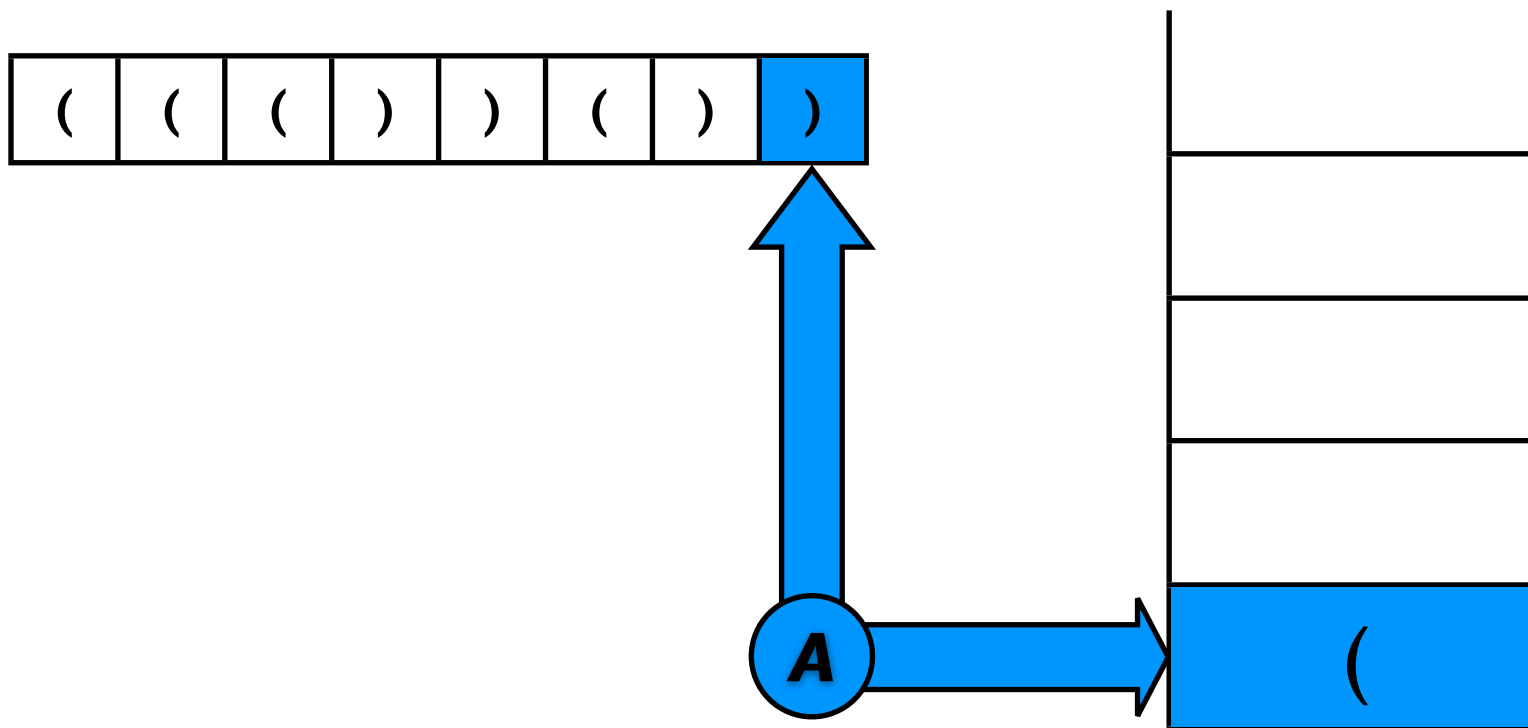
Automates à pile (sans «s»)

Reconnaît les mots correctement parenthésés



Automates à pile (sans «s»)

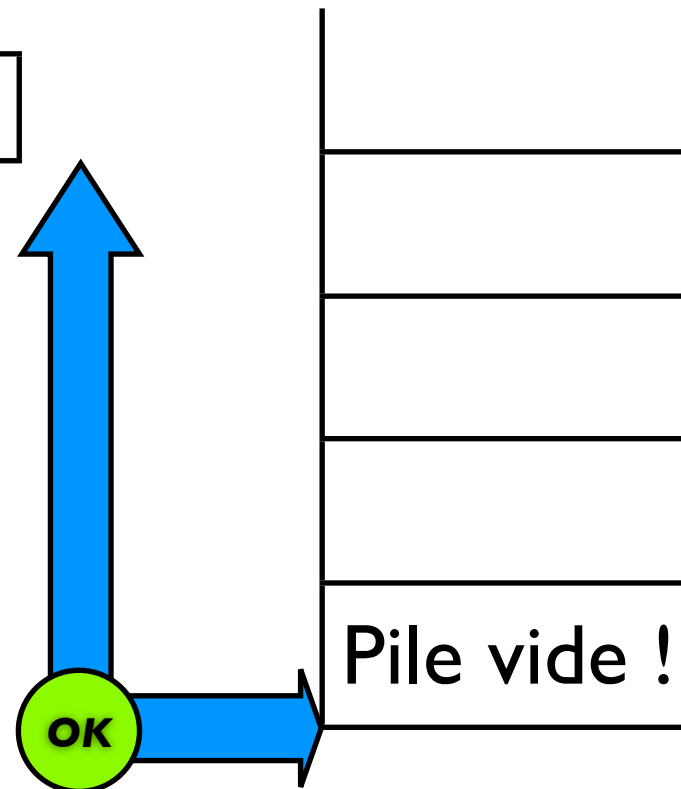
Reconnaît les mots correctement parenthésés



Automates à pile (sans «s»)

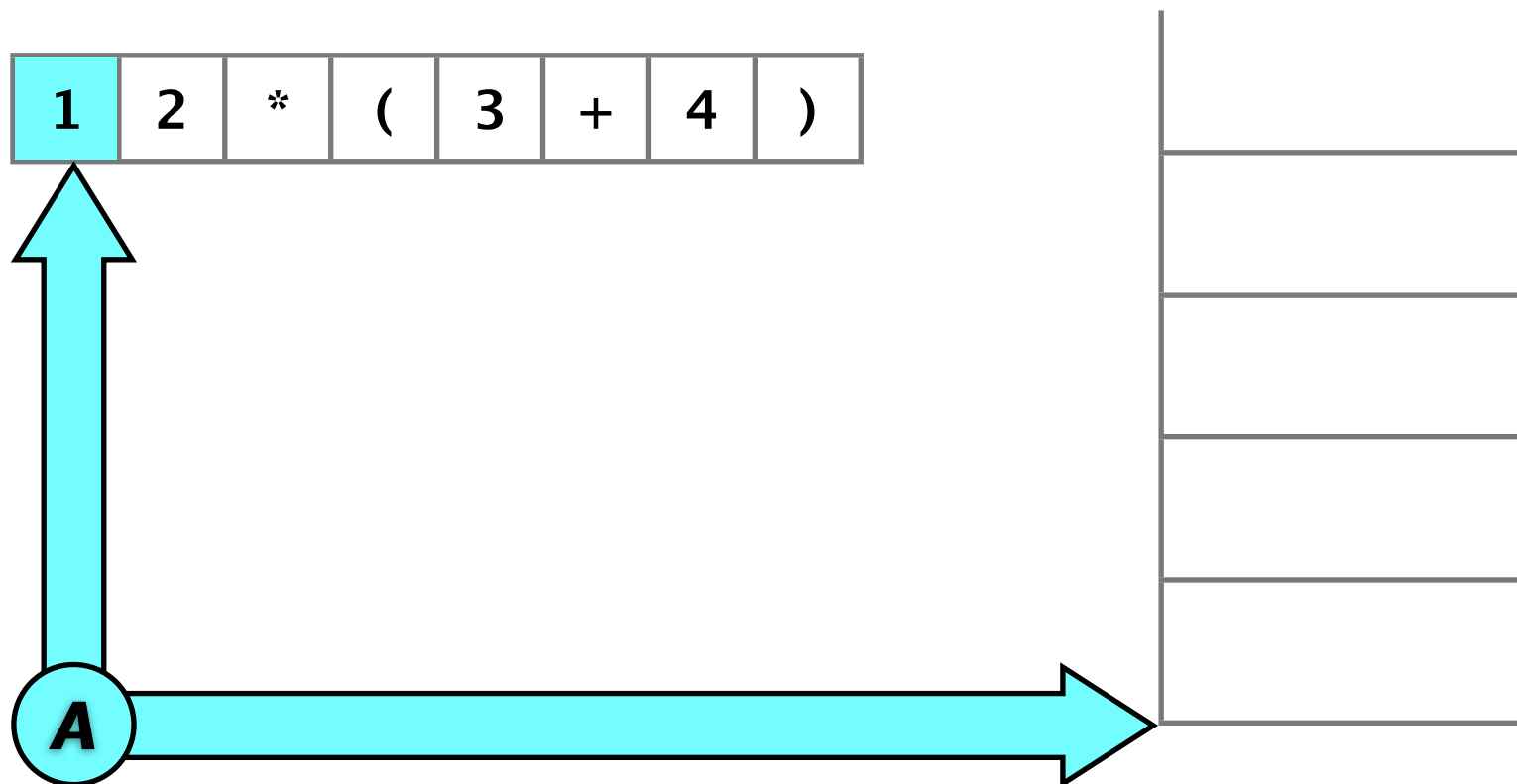
Reconnait les mots correctement parenthésés

((()) ())



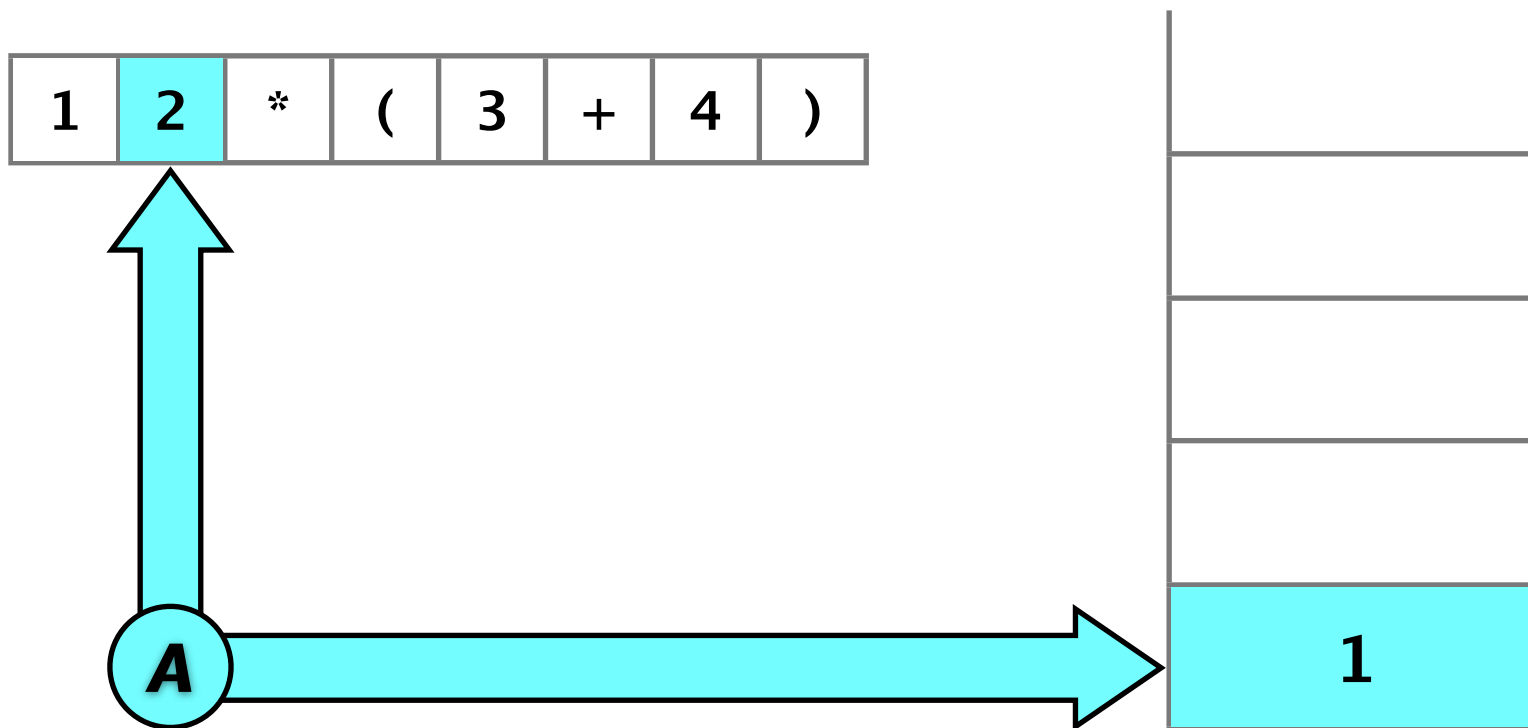
Automates à pile (sans «s»)

Calcule la valeur d'une expression arithmétique



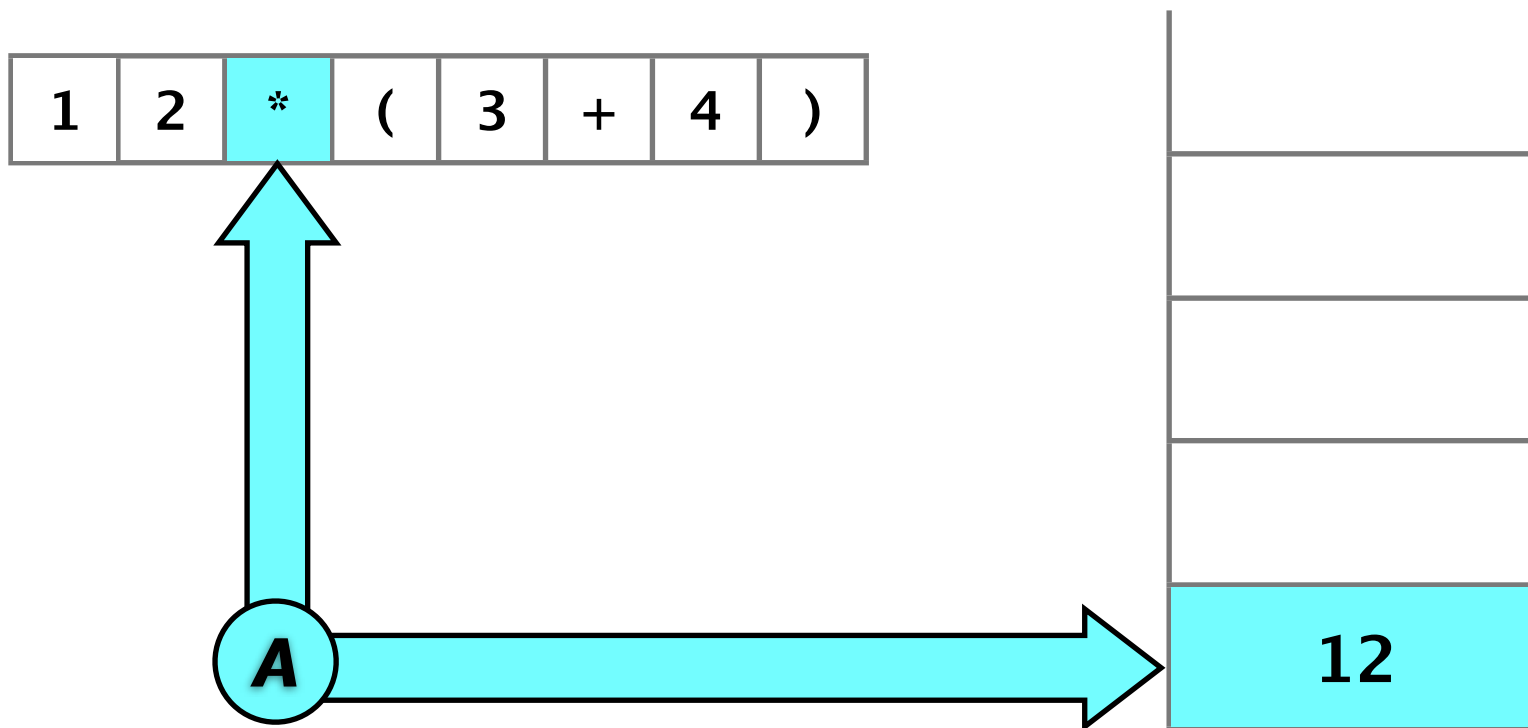
Automates à pile (sans «s»)

Calcule la valeur d'une expression arithmétique



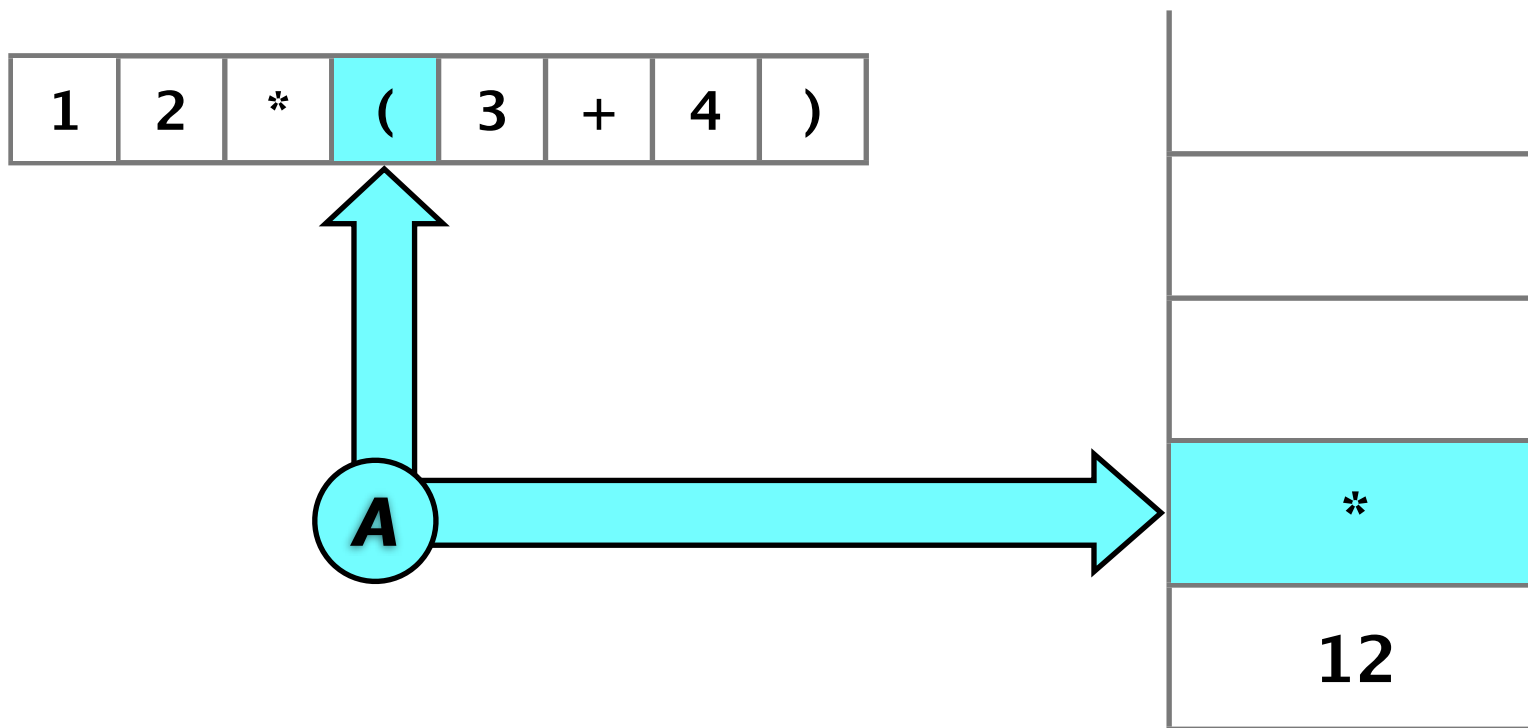
Automates à pile (sans «s»)

Calcule la valeur d'une expression arithmétique



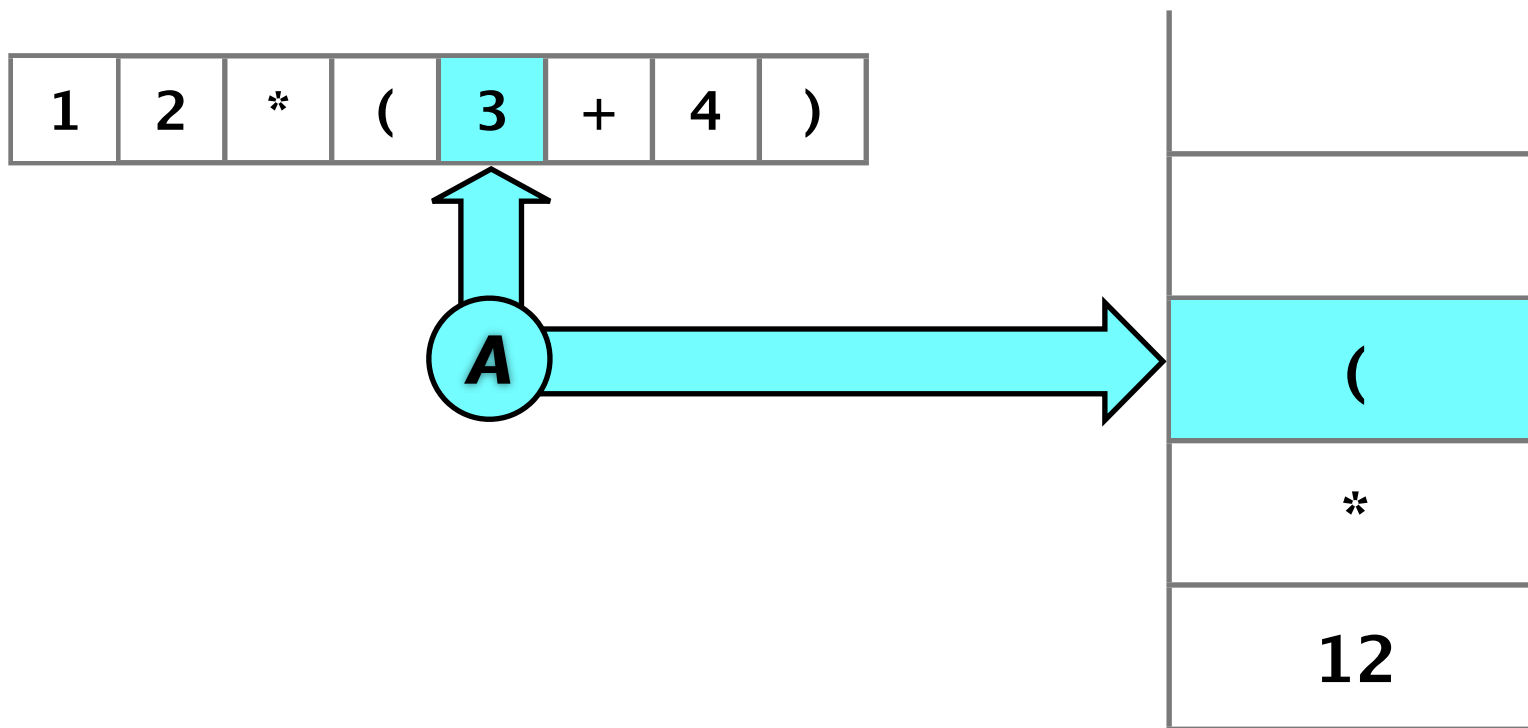
Automates à pile (sans «s»)

Calcule la valeur d'une expression arithmétique



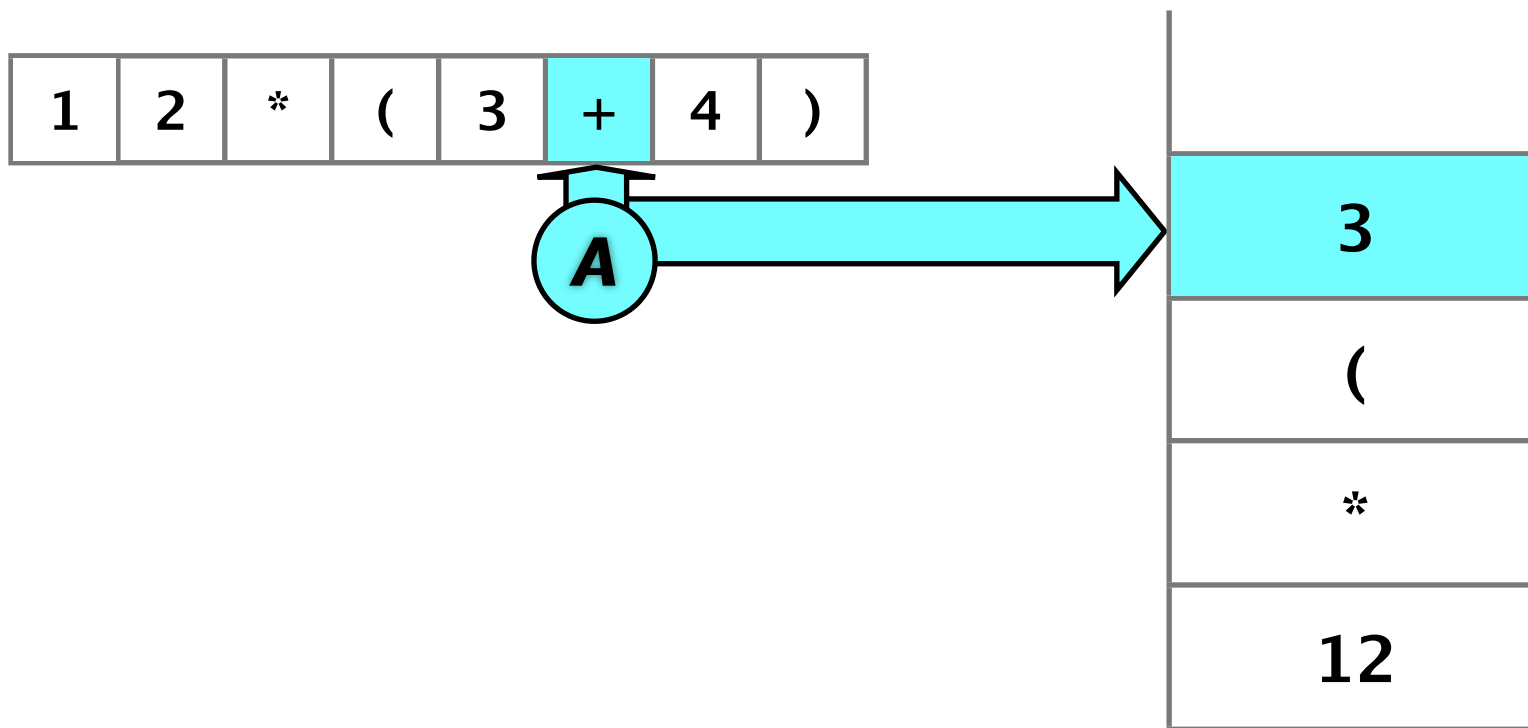
Automates à pile (sans «s»)

Calcule la valeur d'une expression arithmétique



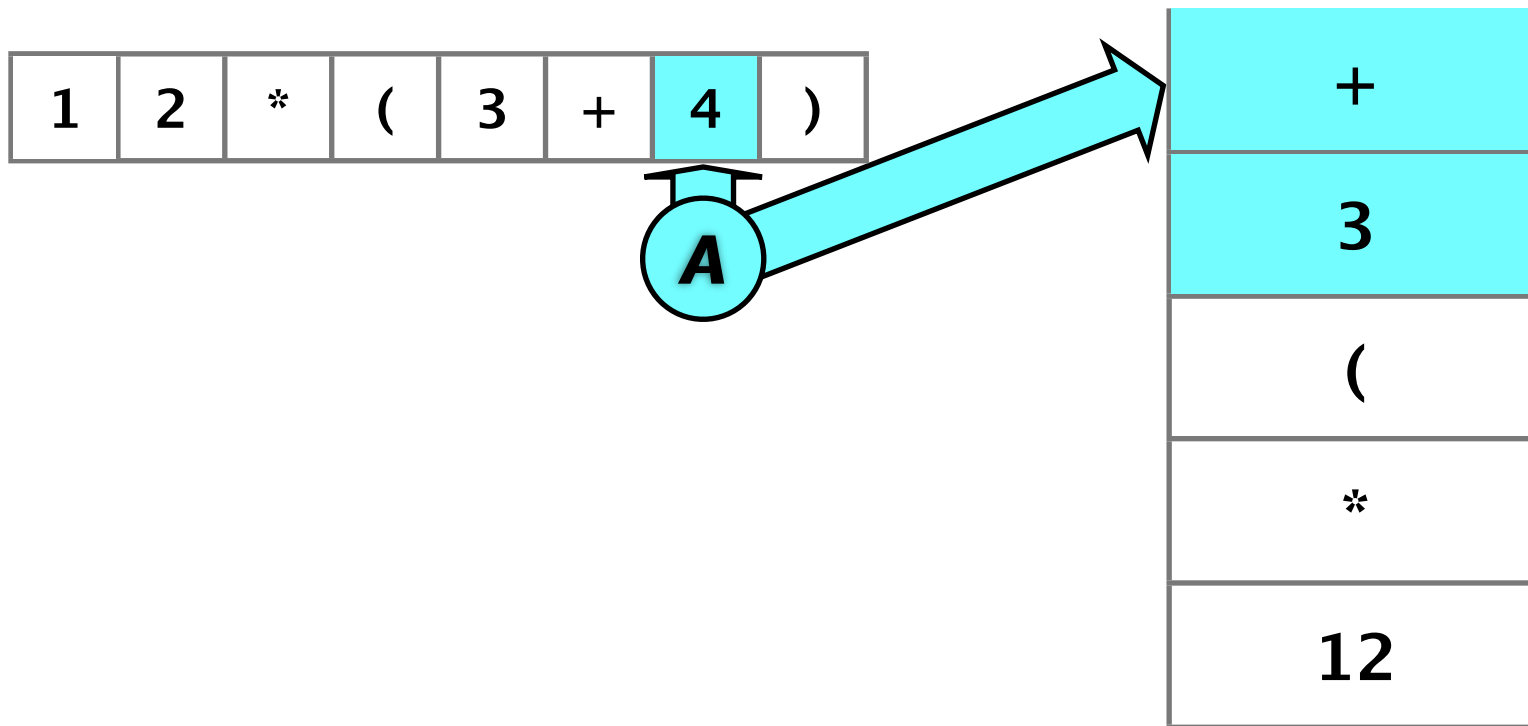
Automates à pile (sans «s»)

Calcule la valeur d'une expression arithmétique



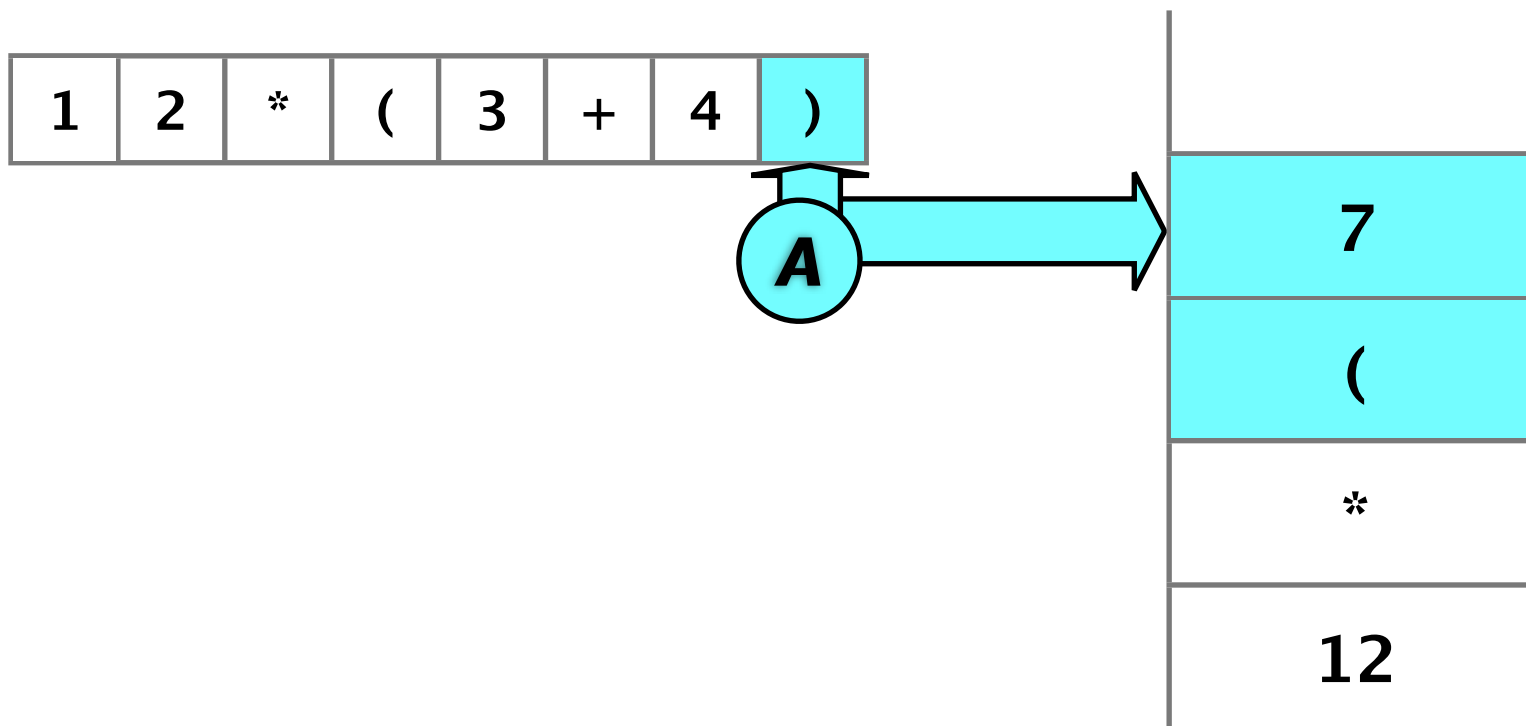
Automates à pile (sans «s»)

Calcule la valeur d'une expression arithmétique



Automates à pile (sans «s»)

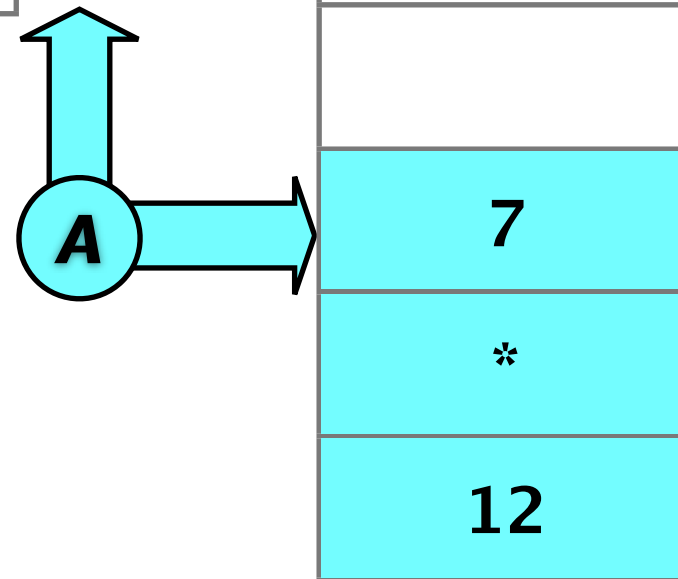
Calcule la valeur d'une expression arithmétique



Automates à pile (sans «s»)

Calcule la valeur d'une expression arithmétique

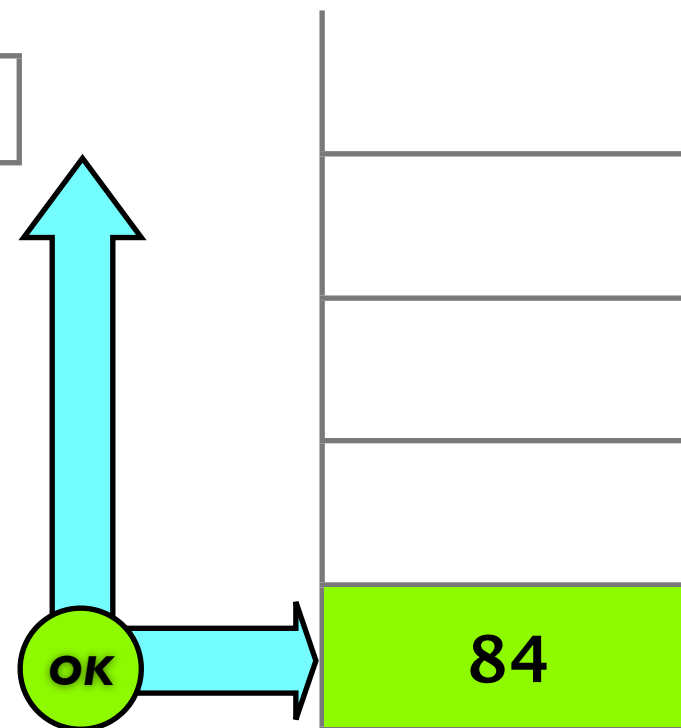
1	2	*	(3	+	4)
---	---	---	---	---	---	---	---



Automates à pile (sans «s»)

Calcule la valeur d'une expression arithmétique

1	2	*	(3	+	4)
---	---	---	---	---	---	---	---



Existe-t-il des mots qui ne soient pas reconnaissables ?

Oui ! Par exemple : $a^n b^n c^n$

Lemme de la pompe (algébrique)

alors $a^{n+p} b^{n+p} c^n$ serait reconnu

Les devises Shadok



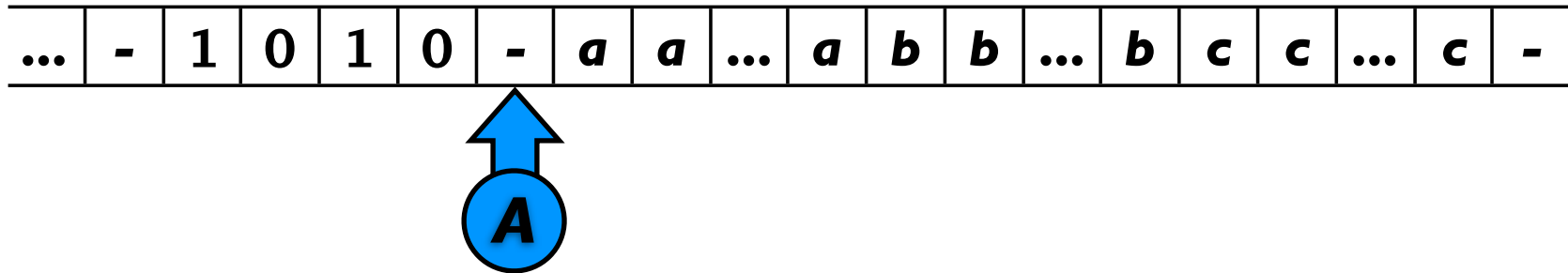
IL VAUT MIEUX POMPER MÊME S'IL NE SE PASSE
RIEN QUE RISQUER QU'IL SE PASSE QUELQUE CHOSE
DE PIÈRE EN NE POMPANT PAS.

Une mémoire potentiellement infinie ne suffit pas !

- ➔ Il faut pouvoir accéder à un emplacement *arbitraire* de la mémoire

RAM = Random Access Memory

Retour aux machines de Turing



Reconnaître $a^n b^n c^n$ avec une Machine de Turing :

- Augmenter un compteur de 1 pour chaque **a** lu,
- puis un autre compteur de 1 pour chaque **b** lu,
- et enfin un dernier compteur de 1 pour chaque **c** lu
- puis de vérifier l'égalité des trois nombres chiffres par chiffres !

➡ **Un nombre fini d'états suffit !**

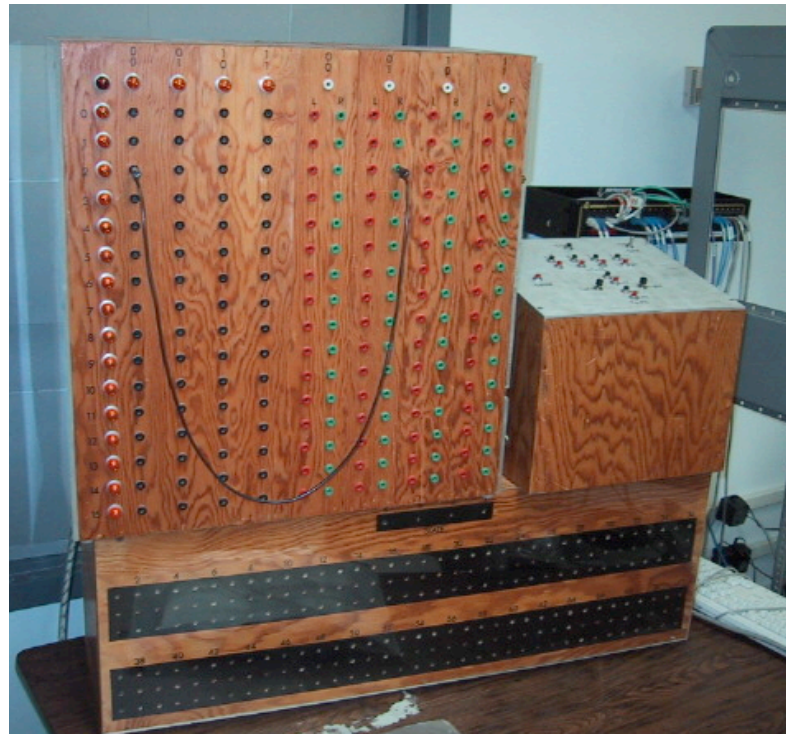
Puissance de calcul des Machines de Turing

Il existe une machine de Turing **universelle** qui
peut simuler **toutes** les machines de Turing,
i.e., que l'on peut programmer pour exécuter
n'importe quel programme

Thèse de Church-Turing (1936)

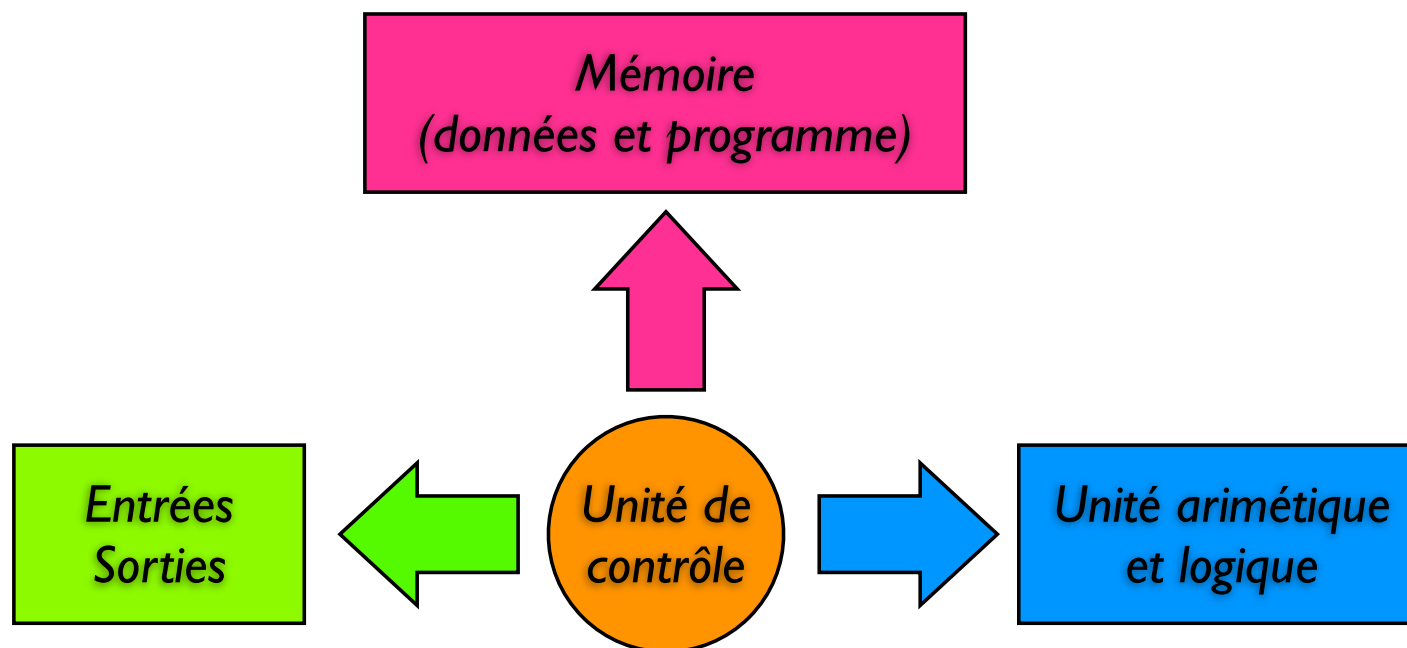
Intérêt des machines de Turing

- Un modèle de calcul pratique pour la théorie
- Performances “similaires” aux ordinateurs réels



Et les ordinateurs alors ?

1944-45 : Architecture de von Neuman, Mauchly & Eckert

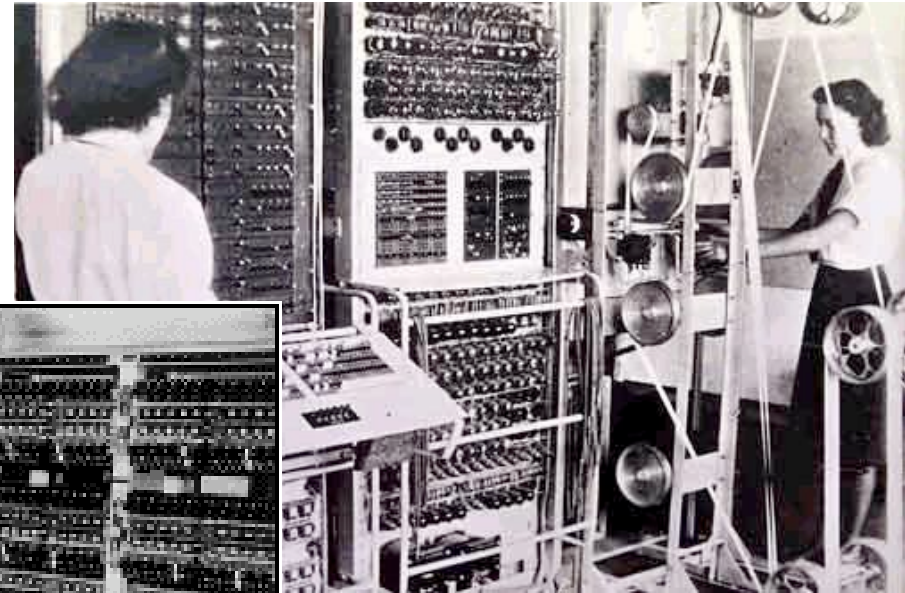


Et les ordinateurs alors ?

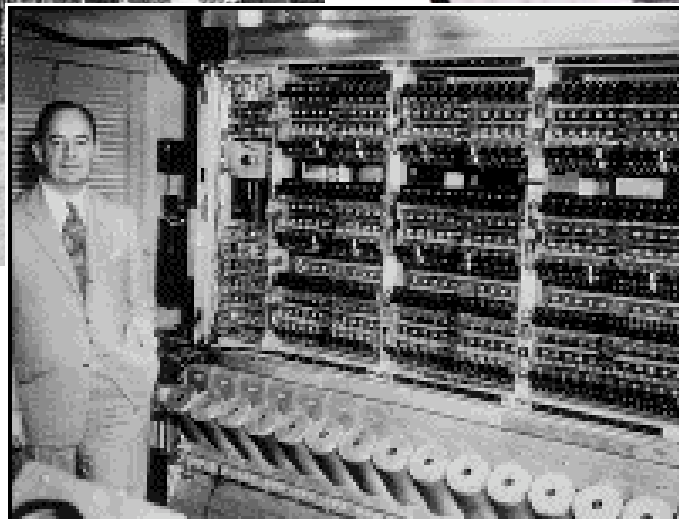
1944-45 : Architecture de von Neuman, Mauchly & Eckert



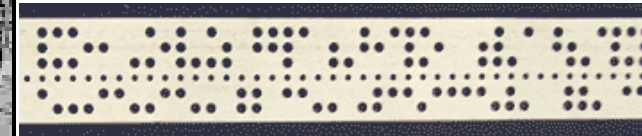
ENIAC (1944)



Colossus (1943)



IAS (1947)



Différents « engins » de calcul

Automates finis :

Reconnaissance de mots et familles de mots (*régulières*)

Automates à pile :

Reconnaissance des langages de programmation (*grammaires*)

Machines de Turing :

Calculent **tout** ce qui est **calculable**

Hiérarchie de Chomsky (1955-65)

**Mais existe-t-il des choses
non-calculables ?**

OUI !

***Mais ça nous emmènerait trop loin
pour aujourd'hui !***

Merci !