Chapitre 1 : bases de la programmation

Fonctions

Nous allons programmer essentiellement des « fonctions ». Une fonction est définie par :

Fonctions

Nous allons programmer essentiellement des « fonctions ». Une fonction est définie par :

- Un ensemble, appelé ensemble de départ, ou domaine de définition. Un élément de l'ensemble de départ s'appelle un « argument » ou « une entrée ».

Fonctions

Nous allons programmer essentiellement des « fonctions ». Une fonction est définie par :

- Un ensemble, appelé ensemble de départ, ou domaine de définition. Un élément de l'ensemble de départ s'appelle un « argument » ou « une entrée ».
- Un autre ensemble, appelé ensemble d'arrivée.

Fonctions

Nous allons programmer essentiellement des « fonctions ». Une fonction est définie par :

- Un ensemble, appelé ensemble de départ, ou domaine de définition. Un élément de l'ensemble de départ s'appelle un « argument » ou « une entrée ».
- Un autre ensemble, appelé ensemble d'arrivée.
- Un moyen d'associer à chaque argument un élément de l'ensemble d'arrivée.

Fonctions

Nous allons programmer essentiellement des « fonctions ». Une fonction est définie par :

- Un ensemble, appelé ensemble de départ, ou domaine de définition. Un élément de l'ensemble de départ s'appelle un « argument » ou « une entrée ».
- Un autre ensemble, appelé ensemble d'arrivée.
- Un moyen d'associer à chaque argument un élément de l'ensemble d'arrivée.

Fonctions

Nous allons programmer essentiellement des « fonctions ». Une fonction est définie par :

- Un ensemble, appelé ensemble de départ, ou domaine de définition. Un élément de l'ensemble de départ s'appelle un « argument » ou « une entrée ».
- Un autre ensemble, appelé ensemble d'arrivée.
- Un moyen d'associer à chaque argument un élément de l'ensemble d'arrivée. Cet élément doit être bien défini, c'est-à-dire exister et être unique.

Fonctions

Nous allons programmer essentiellement des « fonctions ». Une fonction est définie par :

- Un ensemble, appelé ensemble de départ, ou domaine de définition. Un élément de l'ensemble de départ s'appelle un « argument » ou « une entrée ».
- Un autre ensemble, appelé ensemble d'arrivée.
- Un moyen d'associer à chaque argument un élément de l'ensemble d'arrivée. Cet élément doit être bien défini, c'est-à-dire exister et être unique. Si x est un argument, l'élément associé à x s'appelle « l'image » de x, ou la « sortie associée à x ».

Fonctions

Nous allons programmer essentiellement des « fonctions ». Une fonction est définie par :

- Un ensemble, appelé ensemble de départ, ou domaine de définition. Un élément de l'ensemble de départ s'appelle un « argument » ou « une entrée ».
- Un autre ensemble, appelé ensemble d'arrivée.
- Un moyen d'associer à chaque argument un élément de l'ensemble d'arrivée. Cet élément doit être bien défini, c'est-à-dire exister et être unique. Si x est un argument, l'élément associé à x s'appelle « l'image » de x, ou la « sortie associée à x ».

Voyons la syntaxe, d'abord en maths, puis en Python.

Fonctions calculables

Fonctions calculables

- Plusieurs tentatives de formalisation de la notion de fonction "calculable" (début du XX° siècle).

Fonctions calculables

- Plusieurs tentatives de formalisation de la notion de fonction "calculable" (début du XX° siècle).
- Le but est d'exprimer de manière précise le fait qu'on dispose d'une méthode claire, infaillible, et ne nécessitant pas d'astuce pour la calculer.

Fonctions calculables

- Plusieurs tentatives de formalisation de la notion de fonction "calculable" (début du XX° siècle).
- Le but est d'exprimer de manière précise le fait qu'on dispose d'une méthode claire, infaillible, et ne nécessitant pas d'astuce pour la calculer.
- Une telle méthode s'appelle un *algorithme*.

Fonctions calculables

- Plusieurs tentatives de formalisation de la notion de fonction "calculable" (début du XX° siècle).
- Le but est d'exprimer de manière précise le fait qu'on dispose d'une méthode claire, infaillible, et ne nécessitant pas d'astuce pour la calculer.
- Une telle méthode s'appelle un *algorithme*.
- Alonzo Church

Fonctions calculables

- Plusieurs tentatives de formalisation de la notion de fonction "calculable" (début du XX° siècle).
- Le but est d'exprimer de manière précise le fait qu'on dispose d'une méthode claire, infaillible, et ne nécessitant pas d'astuce pour la calculer.
- Une telle méthode s'appelle un *algorithme*.
- Alonzo Church
- Alan Turing (machine de Turing)

Fonctions calculables

- Plusieurs tentatives de formalisation de la notion de fonction "calculable" (début du XX° siècle).
- Le but est d'exprimer de manière précise le fait qu'on dispose d'une méthode claire, infaillible, et ne nécessitant pas d'astuce pour la calculer.
- Une telle méthode s'appelle un *algorithme*.
- Alonzo Church
- Alan Turing (machine de Turing)
- Lambda calcul

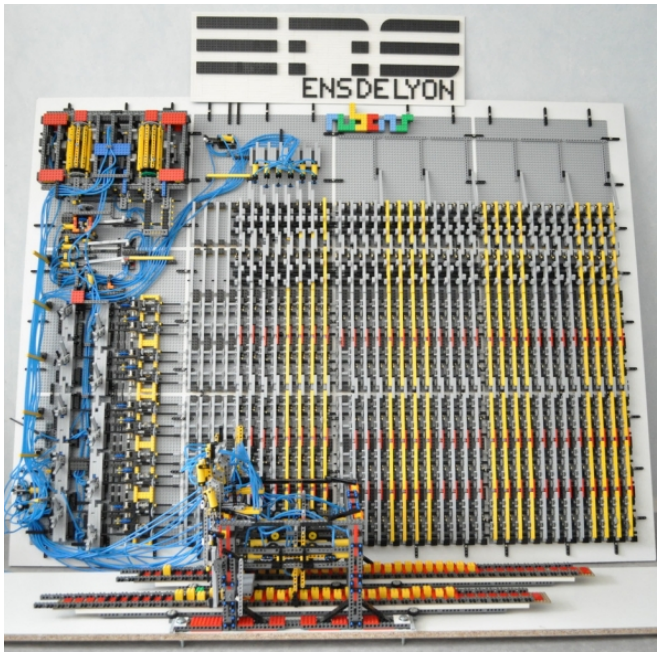
Fonctions calculables

- Plusieurs tentatives de formalisation de la notion de fonction "calculable" (début du XX° siècle).
- Le but est d'exprimer de manière précise le fait qu'on dispose d'une méthode claire, infaillible, et ne nécessitant pas d'astuce pour la calculer.
- Une telle méthode s'appelle un *algorithme*.
- Alonzo Church
- Alan Turing (machine de Turing)
- Lambda calcul
- Les différentes définitions se sont avérées équivalentes !

Ordinateurs

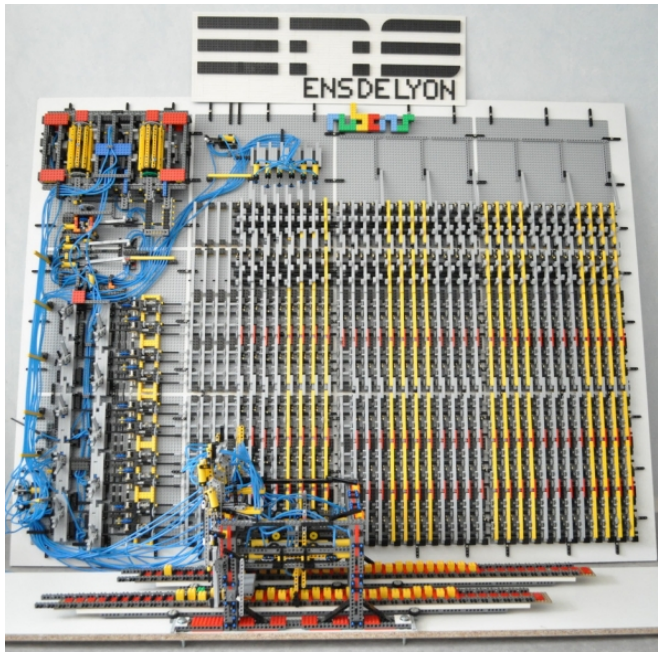
Ordinateurs

- La machine de Turing n'a pas vocation à être effectivement construite.



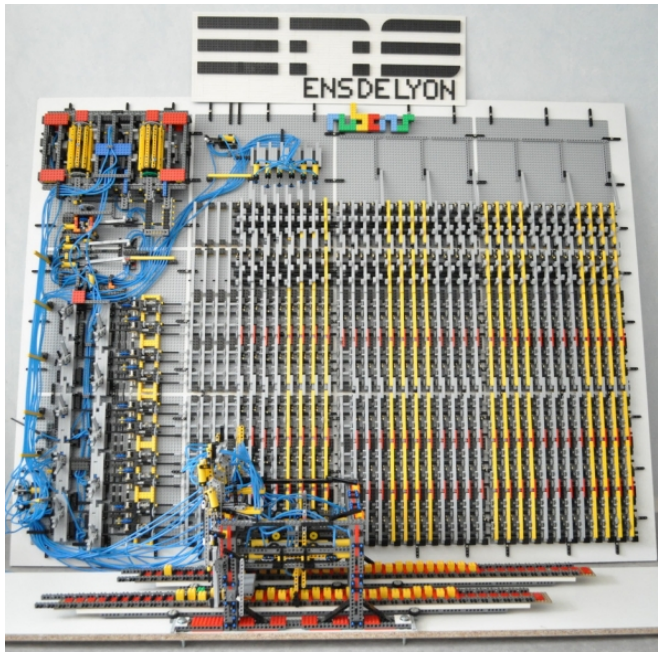
Ordinateurs

- La machine de Turing n'a pas vocation à être effectivement construite.
- Un premier modèle concret d'ordinateur est du à Von Neumann, en 1945.



Ordinateurs

- La machine de Turing n'a pas vocation à être effectivement construite.
- Un premier modèle concret d'ordinateur est du à Von Neumann, en 1945.
- La quasi-totalité des ordinateurs actuels utilisent encore le même principe.



Algorithmes

Algorithmes

Un algorithme est une suite finie d'instructions d'un des types suivants :

Algorithmes

Un algorithme est une suite finie d'instructions d'un des types suivants :

- Enregistrer et lire des données

Algorithmes

Un algorithme est une suite finie d'instructions d'un des types suivants :

- Enregistrer et lire des données
- Effectuer des opérations arithmétiques élémentaires ($+$, $\times$, ...)

Algorithmes

Un algorithme est une suite finie d'instructions d'un des types suivants :

- Enregistrer et lire des données
- Effectuer des opérations arithmétiques élémentaires ($+$, $\times$, ...)
- Passer à l'opération suivante

Algorithmes

Un algorithme est une suite finie d'instructions d'un des types suivants :

- Enregistrer et lire des données
- Effectuer des opérations arithmétiques élémentaires ($+$, $\times$, ...)
- Passer à l'opération suivante
- Effectuer des tests (par exemple avec $<$, $=$, $>$, ...), et choisir l'opération suivante à exécuter en fonction du résultat du test.

Algorithmes

Un algorithme est une suite finie d'instructions d'un des types suivants :

- Enregistrer et lire des données
- Effectuer des opérations arithmétiques élémentaires ($+$, $\times$, ...)
- Passer à l'opération suivante
- Effectuer des tests (par exemple avec $<$, $=$, $>$, ...), et choisir l'opération suivante à exécuter en fonction du résultat du test.
- Répéter des opérations, tant qu'une certaine condition est réalisée.

Langage

Langage

- Pour communiquer avec l'ordinateur, on utilise par un langage de programmation.

Langage

- Pour communiquer avec l'ordinateur, on utilise par un langage de programmation.
- Le programme qui lit votre texte et le traduit pour la machine s'appelle le *compilateur* ou *interpréteur*.

Langage

- Pour communiquer avec l'ordinateur, on utilise par un langage de programmation.
- Le programme qui lit votre texte et le traduit pour la machine s'appelle le *compilateur* ou *interpréteur*.
- Le langage du tronc commun est "Python".

Langage

- Pour communiquer avec l'ordinateur, on utilise par un langage de programmation.
- Le programme qui lit votre texte et le traduit pour la machine s'appelle le *compilateur* ou *interpréteur*.
- Le langage du tronc commun est "Python".
- Le langage de l'option est "Caml".

Langage

- Pour communiquer avec l'ordinateur, on utilise par un langage de programmation.
- Le programme qui lit votre texte et le traduit pour la machine s'appelle le *compilateur* ou *interpréteur*.
- Le langage du tronc commun est "Python".
- Le langage de l'option est "Caml".
- Mais on peut écrire un algorithme aussi en français !

Langage

- Pour communiquer avec l'ordinateur, on utilise par un langage de programmation.
- Le programme qui lit votre texte et le traduit pour la machine s'appelle le *compilateur* ou *interpréteur*.
- Le langage du tronc commun est "Python".
- Le langage de l'option est "Caml".
- Mais on peut écrire un algorithme aussi en français !
- *Remarque :* La plupart des langages viennent avec un grand nombre de programmes préenregistrés (les bibliothèques). Cela peut être un problème pratique que de savoir ce que vous avez le droit d'utiliser lors d'un concours...

Un exemple de fonction non calculable

Un exemple de fonction non calculable

Soit $\mathcal{A}$ l'ensemble de tous les algorithmes.

Un exemple de fonction non calculable

Soit $\mathcal{A}$ l'ensemble de tous les algorithmes. On définit la fonction « `termine` » ainsi :

Un exemple de fonction non calculable

Soit $\mathcal{A}$ l'ensemble de tous les algorithmes. On définit la fonction « `termine` » ainsi :

$$\texttt{termine} : \begin{array}{ccc} \mathcal{A} & \to & \{\text{vrai, faux}\} \\ a & \mapsto & \left| \begin{array}{l} \text{vrai si } a \text{ termine toujours} \\ \text{faux sinon} \end{array} \right. \end{array}$$

On définit alors la fonction `f` par l'algorithme suivant :

On définit alors la fonction `f` par l'algorithme suivant :

```
si termine(f):
    tant que 1==1:
        afficher "bonjour"
sinon:
    renvoyer 247
```

On définit alors la fonction `f` par l'algorithme suivant :

```
si termine(f):
    tant que 1==1:
        afficher "bonjour"
sinon:
    renvoyer 247
```

La fonction `f` termine-t-elle ou non ?

l'absurde

Que faire lorsqu'on aboutit à une conclusion impossible ?

Opérations élémentaires

Voyons comment rédiger les opérations élémentaires, en langage français, puis en Python.

Opérations élémentaires

Voyons comment rédiger les opérations élémentaires, en langage français, puis en Python.

- affectations

Opérations élémentaires

Voyons comment rédiger les opérations élémentaires, en langage français, puis en Python.

- affectations
- tests

Opérations élémentaires

Voyons comment rédiger les opérations élémentaires, en langage français, puis en Python.

- affectations
- tests
- boucles

Les boucles 1

Syntaxe en français :

Tant que *condition* **:**

à répéter tant que la condition est vérifiée

Et en Python :

while *condition* **:**

à répéter tant que la condition est vérifiée

Les boucles 2

Les boucles 2

Souvent on sait à l'avance combien de fois il faudra répéter les opérations. Il existe alors un raccourci très pratique, permettant à la fois de raccourcir le code, mais aussi de réduire le risque d'erreurs.