Chapitre 11 – Cours

Logique, automatismes et Python

Dans ce chapitre, tu vas apprendre à :

- Raisonner avec quantificateurs et implications
- Écrire des fonctions et manipuler des listes
- Tester un algorithme sur des cas limites

Les mots pour comprendre

Définition – Une implication

L'énoncé $P \Rightarrow Q$ affirme que lorsque P est vraie, Q l'est aussi. Sa réciproque doit être étudiée séparément.

Définition – Une fonction Python

Une fonction reçoit des paramètres, exécute des instructions et peut renvoyer une valeur avec `return`. Un test d'égalité s'écrit `==`.

---

1. Énoncés et quantificateurs

Une proposition mathématique est un énoncé dont on peut étudier la vérité. « Pour tout réel x » impose de vérifier chaque réel autorisé ; un exemple favorable ne suffit pas à le démontrer.

Pour réfuter un énoncé universel, un seul contre-exemple suffit. « Il existe un réel x » demande de fournir au moins une valeur qui convient, ou une preuve d'existence.

La négation de « tous les objets ont P » est « au moins un objet n'a pas P ». La négation de « au moins un objet a P » est « aucun objet n'a P ».

On précise toujours l'ensemble étudié : une affirmation sur les naturels peut devenir fausse sur les réels.

2. Implication, réciproque et équivalence

L'implication $P \Rightarrow Q$ affirme que lorsque P est vraie, Q l'est aussi. Elle ne dit pas que Q suffit à garantir P.

La réciproque est $Q \Rightarrow P$; elle doit être examinée séparément. Une équivalence $P \Leftrightarrow Q$ rassemble les deux implications. **Exemple.** Si $x = 2$, alors $x^2 = 4$.

La réciproque est fausse sur les réels car $x = -2$ vérifie aussi $x^2 = 4$. Une résolution d'équation par équivalences conserve exactement l'ensemble des solutions.

Lorsqu'une transformation ne donne qu'une implication, une vérification finale des candidats est indispensable. Les mots « et » et « ou » correspondent respectivement à l'intersection et à la réunion de conditions.

Le « ou » mathématique est généralement inclusif : les deux conditions peuvent être vraies en même temps.

3. Variables, tests et fonctions Python

Une affectation associe une valeur à un nom : `x = 3`. Le test d'égalité s'écrit `x == 3`.

Confondre les deux empêche de comprendre le rôle d'une instruction. Une fonction reçoit des paramètres et renvoie une valeur avec `return`.

Pour une expression mathématique, on précise le domaine autorisé et l'on contrôle les dénominateurs avant de diviser.

```
def image(x):
    return x*x - 3*x + 2
```

Cette fonction représente un calcul d'image. Elle ne résout pas automatiquement une équation et n'étudie pas les variations.

Une approximation décimale de machine peut introduire des écarts : les comparaisons de réels calculés demandent parfois une tolérance.

4. Listes et répétitions

Une liste conserve une suite de valeurs, avec des indices commençant à zéro. Pour `L = [4, 7, 10]`, `L[0]` vaut 4 et `len(L)` vaut 3.

Une boucle `for` parcourt un ensemble fini d'indices. `range(n)` produit les entiers de 0 à $n-1$, et non jusqu'à n inclus. Une boucle `while` répète des instructions tant qu'une condition est

vraie.

Pour un calcul de seuil, on initialise la suite et l'indice, on met à jour les deux, puis on renvoie l'indice de la première valeur qui satisfait la demande. On explique pourquoi l'arrêt est attendu ; une borne de sécurité ne constitue pas une preuve mathématique de terminaison.

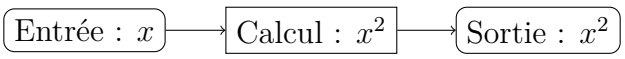
5. Tester et justifier

On confronte un programme à des exemples calculés à la main, à des valeurs limites et à des cas où la réponse change.

Un code qui s'exécute sans erreur peut néanmoins répondre à une autre question que celle posée. La preuve d'une propriété générale ne peut pas être remplacée par un grand nombre de tests.

Les automatismes numériques et algébriques servent à prévoir les ordres de grandeur : un pourcentage, une variation réciproque, un signe ou une solution évidente permettent de contrôler une sortie. Dans tous les cas, on relie le nom des variables à leur signification, on conserve les unités et l'on distingue résultat exact, approximation et conjecture.

Voir la notion



Le schéma distingue ce que l'on reçoit, ce que l'on calcule et ce que l'on renvoie.

Deux exemples, pas à pas

Exemple résolu – 1 – Ou

Question. Résoudre $x < -2$ ou $x > 5$.

Résolution. La réunion est $]-\infty;-2[\cup]5;+\infty[$.

Exemple résolu – 2 – Somme

Question. Calculer la valeur finale de S, initialisée à zéro, si l'on ajoute successivement les entiers de 1 à 5.

Résolution. $S = 1 + 2 + 3 + 4 + 5 = 15$.