Leçon 9 - Calculs et opérations avec $\sum$ (ou $\prod$)

⇒ Principes de démonstration

⇒ Deux raisonnements

⇒ **Reprendre quelques principes de base de démonstration**

⇒ **Deux raisonnements en particulier**

⇒ **Reprendre quelques principes de base de démonstration**

⇒ **Deux raisonnements en particulier**

Confusion fréquente. . .un peu de bon sens

Il s'agit simplement *d'un peu de bon sens* sur la signification des mots « nécessaire » et « suffisant ».

Exemple A : « x est le carré d'un entier »et B : « $x \geqslant 0$ »

Qu'est-ce qui est nécessaire, qu'est-ce qui est suffisant (ou ne l'est pas) ?

Confusion fréquente. . .un peu de bon sens

Il s'agit simplement *d'un peu de bon sens* sur la signification des mots « nécessaire » et « suffisant ».

Exemple A : « x est le carré d'un entier »et B : « $x \geqslant 0$ »

Qu'est-ce qui est nécessaire, qu'est-ce qui est suffisant (ou ne l'est pas) ?

Définition - Condition nécessaire. Condition suffisante

Plus généralement $A \Rightarrow B$ peut se dire :

- B est une condition nécessaire (**CN**) pour avoir A (puisque si on A, nécessairement on a B)
- A est une condition suffisante (**CS**) pour avoir B (puisqu'il suffit d'avoir A pour avoir B

Rechercher une condition nécessaire et suffisante (**CNS**) pour avoir A revient donc à chercher B tel que $A \Leftrightarrow B$.

Analyse-Synthèse

Savoir-faire. Analyse-Synthèse

Certaines démonstrations, difficiles à gérer par équivalences, ou lorsque le résultat n'est pas donné, se font en deux phases :

1. phase d' "analyse" : on obtient une condition nécessaire (par implications successives par exemple) pour qu'une première hypothèse soit vérifiée
2. phase de "synthèse", ou phase de vérification : la condition précédemment obtenue est-elle suffisante ?

Analyse-Synthèse

Savoir-faire. Analyse-Synthèse

Certaines démonstrations, difficiles à gérer par équivalences, ou lorsque le résultat n'est pas donné, se font en deux phases :

1. phase d' "analyse" : on obtient une condition nécessaire (par implications successives par exemple) pour qu'une première hypothèse soit vérifiée
2. phase de "synthèse", ou phase de vérification : la condition précédemment obtenue est-elle suffisante ?

Exercice

Montrer que toute fonction définie sur $\mathbb{R}$ à valeurs dans $\mathbb{R}$ s'écrit de manière unique comme somme d'une fonction paire et d'une fonction impaire.

Analyse-Synthèse

Exercice

Soit A, B, C et D quatre points du plan tel que $AC = BD$ et (AB) non parallèle à (CD).

Alors il existe une unique rotation du plan r tel que $r(A) = B$ et $r(C) = D$.

Donner ses caractérisations

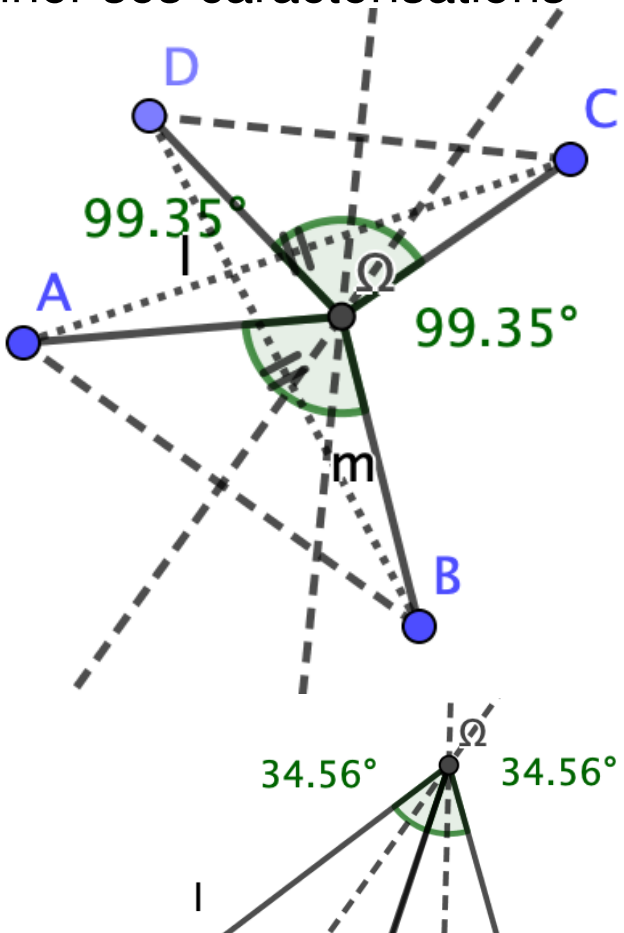
Analyse-Synthèse

Exercice

Soit A, B, C et D quatre points du plan tel que $AC = BD$ et (AB) non parallèle à (CD).

Alors il existe une unique rotation du plan r tel que $r(A) = B$ et $r(C) = D$.

Donner ses caractérisations



⇒ **Reprendre quelques principes de base de démonstration**

⇒ **Deux raisonnements en particulier**

Principe

Savoir-faire. Utilisation d'un contre-exemple

Lorsque l'on veut prouver qu'une implication est fausse, on cherche un exemple vérifiant l'hypothèse mais pas la conclusion, ce que l'on appelle un *contre-exemple*.

Principe

Savoir-faire. Utilisation d'un contre-exemple

Lorsque l'on veut prouver qu'une implication est fausse, on cherche un exemple vérifiant l'hypothèse mais pas la conclusion, ce que l'on appelle un *contre-exemple*.

Exercice

Soit f la fonction définie sur $\mathbb{R}$ par $f(x) = |x + \frac{3}{2}| - \frac{1}{2}$. Montrer que $(x \geqslant -1)$ et $(f(x) \geqslant 0)$ ne sont pas équivalents.

⇒ **Reprendre quelques principes de base de démonstration**

⇒ **Deux raisonnements en particulier**

Principe

Principe admis (axiome)

Soit $P(n)$ (parfois notée $\mathscr{P}_n$) une propriété portant sur l'entier n.
Si on a

$$\begin{cases} P(0) \text{ vraie} \\ \forall n \in \mathbb{N}, (P(n) \text{ vraie } \Rightarrow P(n+1) \text{ vraie }) \end{cases}$$

alors $P(n)$ est vraie pour tout entier n.

Principe

Principe admis (axiome)

Soit $P(n)$ (parfois notée $\mathscr{P}_n$) une propriété portant sur l'entier n.
Si on a

$$\begin{cases} P(0) \text{ vraie} \\ \forall n \in \mathbb{N}, (P(n) \text{ vraie } \Rightarrow P(n+1) \text{ vraie }) \end{cases}$$

alors $P(n)$ est vraie pour tout entier n.

Un idée de démonstration ?
(A quoi est équivalent le principe de récurrence ?)

Rédaction

Savoir-faire. Rédaction d'un raisonnement par récurrence

- Pour $n = 0$, $P(0)$ est vérifiée, avec vérification effective ! (souvent deux simples calculs)
- Supposons la propriété vérifiée pour **un certain** $n \geqslant 0$ (et surtout pas "pour tout", parce que là, ce n'est plus la peine de faire une démonstration !).

    $\Rightarrow$ Montrons que $P(n+1)$ est vraie.

Conclusion : $\forall n \in \mathbb{N}, P(n)$ est vraie.

Quelques remarques

Remarque Démarrer à un autre nombre

Quelques remarques

Remarque Démarrer à un autre nombre

Exercice

Montrer que pour tout $n \in \mathbb{N}$, $n < 2^n$.

Quelques remarques

Remarque Démarrer à un autre nombre

Exercice

Montrer que pour tout $n \in \mathbb{N}$, $n < 2^n$.

Exercice

Y a-t-il des erreurs dans les raisonnements suivants ? Où sont-elles ?

Récurrences plus subtiles

Savoir-faire. Récurrence à plusieurs pas (ou plusieurs termes)

Suivant la façon dont est énoncée la propriété de récurrence $P(n)$ il peut être nécessaire

- d'initialiser la récurrence avec plusieurs (k) valeurs de n
- de supposer $P(n), \ldots, P(n+k-1)$ (il y en a aussi k) vraies

pour un certain $n \geqslant 0$

- de prouver que ces k propriétés exactes entraînent $P(n+k)$ vraie

Récurrences plus subtiles

Savoir-faire. Récurrence à plusieurs pas (ou plusieurs termes)

Suivant la façon dont est énoncée la propriété de récurrence $P(n)$ il peut être nécessaire

- d'initialiser la récurrence avec plusieurs (k) valeurs de n
- de supposer $P(n),\dots,P(n+k-1)$ (il y en a aussi k) vraies

pour un certain $n \geqslant 0$

- de prouver que ces k propriétés exactes entraînent $P(n+k)$ vraie

Savoir-faire. Récurrence forte

- Pour $n=0$, $P(0)$ est vérifiée (avec vérification effective !)
- Supposons la propriété vérifiée **jusqu'à** un certain $n \geqslant 0$ (i.e $P(0),P(1),\dots,P(n)$ vraies)
- Montrons que $P(n+1)$ est vraie.

Exercices

Exercice

Soit (u_n) la suite définie par $u_0 = \frac{2}{5}, u_1 = 1$ et pour tout entier naturel n,

$$u_{n+2} = 5u_{n+1} - 6u_n.$$

Démontrer que pour tout $n \in \mathbb{N}$, $u_n = \frac{2^n + 3^n}{5}$.

Exercices

Exercice

Soit (u_n) la suite définie par $u_0 = \frac{2}{5}, u_1 = 1$ et pour tout entier naturel n,

$$u_{n+2} = 5u_{n+1} - 6u_n.$$

Démontrer que pour tout $n \in \mathbb{N}$, $u_n = \frac{2^n + 3^n}{5}$.

Exercice

Montrer que tout entier $n \geqslant 2$ se décompose en produit de nombres premiers.

Exercices

Exercice

Soit (u_n) la suite définie par $u_0 = \frac{2}{5}, u_1 = 1$ et pour tout entier naturel n,

$$u_{n+2} = 5u_{n+1} - 6u_n.$$

Démontrer que pour tout $n \in \mathbb{N}$, $u_n = \frac{2^n + 3^n}{5}$.

Exercice

Montrer que tout entier $n \geqslant 2$ se décompose en produit de nombres premiers.

Exercice

Montrer qu'un changement d'hypothèse de récurrence ramène une récurrence à plusieurs pas ou une récurrence forte à une récurrence faible.

⇒ **Reprendre quelques principes de base de démonstration**

⇒ **Deux raisonnements en particulier**

Algorithme

Heuristique. Exploitation d'un algorithme

Un algorithme peut permettre de démontrer, constructivement, l'existence d'un certain objet (ou d'une certaine fonction).
La difficulté est plutôt de démontrer que l'algorithme :

- se termine bien
- réalise bien ce que l'on désire

Algorithme

Heuristique. Exploitation d'un algorithme

Un algorithme peut permettre de démontrer, constructivement, l'existence d'un certain objet (ou d'une certaine fonction).
La difficulté est plutôt de démontrer que l'algorithme :

- se termine bien
- réalise bien ce que l'on désire

Définition - Algorithme

Un algorithme est une suite finie de règles à appliquer dans un ordre déterminé à un nombre fini de données pour arriver avec certitude (c'est-à-dire sans indétermination ou sans ambiguïté), en un nombre fini d'étapes, à un certain résultat et cela indépendamment des données.

Terminaison de l'algorithme

Pour démontrer que l'algorithme termine (que le nombre d'étapes est fini), on exploite un variant de boucle, en règle général, en suite d'entiers décroissante. Pour des boucles `for`, la terminaison de boucle est en règle générale immédiate.

Terminaison de l'algorithme

Pour démontrer que l'algorithme termine (que le nombre d'étapes est fini), on exploite un variant de boucle, en règle général, en suite d'entiers décroissante. Pour des boucles `for`, la terminaison de boucle est en règle générale immédiate.

Savoir-faire. Démontrer qu'une boucle se termine

On identifie une expression (variable) qui :

- est entière
- décroît strictement à chaque étape de la boucle

Alors, nécessairement, la boucle se termine.

Terminaison de l'algorithme

Pour démontrer que l'algorithme termine (que le nombre d'étapes est fini), on exploite un variant de boucle, en règle général, en suite d'entiers décroissante. Pour des boucles `for`, la terminaison de boucle est en règle générale immédiate.

Savoir-faire. Démontrer qu'une boucle se termine

On identifie une expression (variable) qui :

- est entière
- décroît strictement à chaque étape de la boucle

Alors, nécessairement, la boucle se termine.

Exemple Division euclidienne

Exercice

Exercice

On considère le bout de programme suivant :

```
c=0
while p>0 :
        if c==0 :
                p=p-2
                c=1
        else :
                p=p+1
                c=0
```

1. Que fait ce programme ?
2. Les variables `p` et `c` sont-elles décroissantes, strictement ?
3. Montrer que la variable `t=2p+3c` est entière, strictement décroissante.
   En déduire la terminaison de l'algorithme.

Correction de l'algorithme

Il faut aussi savoir **démontrer** que le programme (avec de nombreuses répétitions de la boucle) réalise se que l'on souhaite. On utilise alors pour cela des *invariant de boucles*.

Correction de l'algorithme

Il faut aussi savoir **démontrer** que le programme (avec de nombreuses répétitions de la boucle) réalise se que l'on souhaite. On utilise alors pour cela des *invariant de boucles*.

Savoir-faire. Usage d'un invariant de boucle pour démontrer le résultat attendu

Pour démontrer qu'une boucle réalise bien le résultat attendu,

1. on cherche une expression qui reste constante tout au long des calculs de la boucle ;
2. on calcule sa valeur initiale (avant le début de la boucle) ;
3. on en déduit sa valeur finale ;
4. enfin connaissant les valeurs finales des variables du système (testées pour la sortie de boucle), on en déduit la valeur de la variable retournée par le programme.

Exemple d'invariant de boucle

Exemple Retour sur la division euclidienne

Exemple d'invariant de boucle

Exemple Retour sur la division euclidienne

Exercice

Montrer que le programme

```
1  def somme2(n) :
2      S=0
3      for i in range(1,n+1):
4          S=S+i**2
5      return(S)
```

calcule bien $\sum_{i=1}^{100} i^2$

Exemple d'invariant de boucle

Exemple Retour sur la division euclidienne

Exercice

Montrer que le programme

```
1  def somme2(n) :
2      S=0
3      for i in range(1,n+1):
4          S=S+i**2
5      return(S)
```

calcule bien $\sum_{i=1}^{100} i^2$

Exercice

On cherche à écrire un programme qui calcul $n!$.

1. Ecrire un programme avec une boucle `while`.
2. **Démontrer** que le programme se termine bien.
3. **Démontrer** que le programme effectue bien ce que l'on souhaite.

Conclusion

Objectifs

⇒ Reprendre quelques principes de base de démonstration

⇒ Deux raisonnements en particulier

Conclusion

Objectifs

⇒ Reprendre quelques principes de base de démonstration

- Implications, double implication et équivalence

Conclusion

Objectifs

⇒ Reprendre quelques principes de base de démonstration

- Implications, double implication et équivalence
- C.N. et C.S. Méthode analyse-synthèse.

Conclusion

Objectifs

⇒ Reprendre quelques principes de base de démonstration

- Implications, double implication et équivalence
- C.N. et C.S. Méthode analyse-synthèse.
- Contraposée voire raisonnement par l'absurde

Conclusion

Objectifs

⇒ Reprendre quelques principes de base de démonstration

⇒ Deux raisonnements en particulier

Conclusion

Objectifs

⇒ Reprendre quelques principes de base de démonstration

⇒ Deux raisonnements en particulier

- Récurrence (suite d'assertions impliquées successivement)

Conclusion

Objectifs

⇒ Reprendre quelques principes de base de démonstration

⇒ Deux raisonnements en particulier

- Récurrence (suite d'assertions impliquées successivement)
- Utilisation d'un algorithme

Conclusion

Objectifs

⇒ Reprendre quelques principes de base de démonstration

⇒ Deux raisonnements en particulier

Pour la prochaine fois

- Lecture du cours : chapitre 3. Trigonométrie
- Exercice n°238 & 2239