Réunion de rentrée DL1

Pierre Rousselin

Université Paris 13 – Institut Galilée

1er septembre 2022

Où sommes-nous ?

Double-licence

Personnel de l'université

Réussir en DL1

Un problème Maths – Info

Où sommes-nous ?

Double-licence

Personnel de l'université

Réussir en DL1

Un problème Maths – Info

Université Paris XIII

- Université Paris XIII dénommée « Sorbonne Paris Nord », créée en 1970
- Campus de Villetaneuse, petite ville de 13000 habitants en Seine-Saint-Denis
- Institut Galilée, regroupe la recherche et l'enseignement en sciences « dures » de l'université Paris 13
- Parmi les laboratoires de recherche :
  - LAGA, Laboratoire Analyse, Géométrie et Applications, le « labo de maths »
  - LIPN, Laboratoire d'Informatique de Paris Nord, le « labo d'info »
  - L2TI, Laboratoire de Traitement et Transport de l'Information

Parmi les formations de l'institut Galilée

- licences de maths, d'info et double-licence.
- masters :
  - informatique, parcours Programmation et Logiciels Sûrs (PLS) et Exploration Informatique des Données et Décisionnel (EI2D)
  - mathématiques fondamentales
  - enseignement des mathématiques (préparation au CAPES)
  - mathématiques des données
- école d'ingénieur « Sup'Galilée », notamment parcours :
  - info et info en apprentissage
  - MACS (mathématiques appliquées au calcul scientifique)
- école doctorale
- école universitaire de recherche (EUR) « *Mathematics and Computer Science* » bourses de master ou doctorat pour des thèmes maths-info.

Où sommes-nous ?

Double-licence

Personnel de l'université

Réussir en DL1

Un problème Maths – Info

Double-licence = deux licences

- Vous êtes *à la fois* en licence d'informatique *et* en licence de mathématiques.
- Idéalement, vous décrochez donc deux diplômes de licence en 3 ans, soit 6 semestres ou « niveaux. »
- Chaque semestre, idéalement vous validez un niveau de licence de maths et un niveau de licence d'info.
- Chaque semestre, vous pouvez, par choix ou parce qu'on vous l'impose basculer en licence « simple » de maths ou en licence « simple » d'info.

Organisation des 3 années

- L1 Peu de différence avec la L1 maths et la L1 info. But du jeu pour vous : acquérir des bases très très solides pour pouvoir tenir le coup plus tard. Viser plus de 14 de moyenne.
- L2 Ça se complique beaucoup, car la L2 maths et la L2 info divergent donc plus de cours à suivre, beaucoup plus de travail à fournir.
- L3 Premier semestre ici, toujours difficile, second semestre, *a priori* à l'international.

Niveau 1 (N1)

UE (Unités d'Enseignements) fondamentales

- algèbre 1 : Introductions aux structures mathématiques. Bases du langage et du raisonnement mathématique. 6 ECTS
- analyse 1 : Suites et fonctions continues. Bases de l'analyse mathématique. 7 ECTS
- programmation 1 : Éléments d'informatique. Initiation à la programmation, en C. 7 ECTS
- initiation à l'environnement Unix. Apprendre à utiliser la ligne de commande Unix, quelques concepts clés de systèmes d'exploitation, écriture de scripts shell. 6 ECTS

UE transversales dans un bloc « culture générale. »

- anglais, 2 ECTS
- bureautique scientifique, 1 ECTS
- initiation aux preuves formelles [1] spécifique double-licence, 1 ECTS

1. qui n'est dans ce bloc que pour de basses raisons administratives...

Niveau 1 (N1)

Le niveau 1 est la base de la base et conditionne toute la suite.

- Si vos résultats au niveau 1 sont juste moyens, il est très probable que vous raterez le niveau 2.
- Si vos résultats au niveau 1 sont vraiment insuffisants, vous ne serez pas autorisé à suivre les cours du niveau 2, mais à la place passerez en parcours aménagé et ne pourrez valider la L1 qu'en deux ans.

Règles précises

Pour valider un niveau, en général, 3 conditions :

- au moins 6 de moyenne à chaque UE fondamentale et
- au moins 10 à la moyenne des UE fondamentales et
- au moins 10 de moyenne générale.

Règles précises

Pour valider un niveau, en général, 3 conditions :

- au moins 6 de moyenne à chaque UE fondamentale et
- au moins 10 à la moyenne des UE fondamentales et
- au moins 10 de moyenne générale.

Pour passer du niveau 1 au niveau 2 :

- Au moins 50% des ECTS des UE fondamentales obtenues (donc, en gros, 2 UE à plus de 10) ;
- Au moins 6 à la moyenne des UE fondamentales.

`https://galilee.univ-paris13.fr/licence/` onglet « modalités du contrôle des connaissances et compétences »

ANNEE UNIVERSITAIRE 2022-2023

LICENCE Licence 1ère année

SEMAINE	29-août 02-sept	05-sept 09-sept	12-sept 16-sept	19-sept 23-sept	26-sept 30-sept	03-oct 07-oct	10-oct 14-oct
Lundi Mardi Mercredi Jeudi Vendredi	Pré-rentrée	1	2	3	4	5	sem. 6 INFO/MATH/DL et Partiels 1 PC / SPI

SEMAINE	17-oct 21-oct	24-oct 28-oct	31-oct 04-nov	07-nov 11-nov	14-nov 18-nov	21-nov 25-nov	28-nov 02-déc
Lundi Mardi Mercredi Jeudi Vendredi	semaine 6 PC / SPI Partiels 1 INFO/MATH/DL	7	VACANCES D'AUTOMNE	8	9	10	11

SEMAINE	05-déc 09-déc	12-déc 16-déc	19-déc 23-déc	26-déc 30-déc	02-janv 06-janv	09-janv 13-janv	16-janv 20-janv
Lundi Mardi Mercredi Jeudi Vendredi	12	Partiels 2 N1 DAPS)	VACANCES DE NOEL			Semaine Orientation, Sciences et Culture JURY N1	Prérequis N2

SEMAINE	23-janv 27-janv	30-janv 03-févr	06-févr 10-févr	13-févr 17-févr	20-févr 24-févr	27-févr 03-mars	06-mars 10-mars
Lundi Mardi Mercredi Jeudi Vendredi	1	2	3	4	5	VACANCES D'HIVER	6

SEMAINE	13-mars 17-mars	20-mars 24-mars	27-mars 31-mars	03-avr 07-avr	10-avr 14-avr	17-avr 21-avr	24-avr 28-avr
Lundi Mardi Mercredi Jeudi Vendredi	7	Partiels 1 N2	8	9	10	11	VACANCES DE PRINTEMPS

SEMAINE	01-mai 05-mai	08-mai 12-mai	15-mai 19-mai	22-mai 26-mai	29-mai 02-juin	05-juin 09-juin	12-juin 16-juin
Lundi Mardi Mercredi Jeudi Vendredi	VACANCES DE PRINTEMPS	12	Rattrapage jours fériés	Partiels 2 N2 (DAPS)	SOUTIEN N1 (commun à PAm)	SOUTIEN N1 (commun à PAm)	2de chance N1 Jury N2

SEMAINE	19-juin 23-juin	26-juin 30-juin	03-juil 07-juil	10-juil 14-juil	17-juil 21-juil
Lundi Mardi Mercredi Jeudi Vendredi	SOUTIEN N2	2de chance N2		Jury 2de chance L1	

Où sommes-nous ?

Double-licence

Personnel de l'université

Réussir en DL1

Un problème Maths – Info

Contacts

Vos interlocuteurs privilégiés cette année :

Secrétariat L1 Galilée Loredana Yacoubi,
`licence1.galilee@univ-paris13.fr`

Responsable double-licence Pierre Rousselin,
`rousselin@univ-paris13.fr`

Contacts

Vos interlocuteurs privilégiés cette année :

Secrétariat L1 Galilée Loredana Yacoubi,
`licence1.galilee@univ-paris13.fr`

Responsable double-licence Pierre Rousselin,
`rousselin@univ-paris13.fr`

Règles à respecter :

- Utilisez vos adresses `@edu.univ-paris13.fr` et consultez-les régulièrement (au moins 1 fois par jour).
- Un **objet** clair, net et précis.
- Dans le corps,
  - rappelez qui vous êtes ;
  - faîtes des phrases courtes et compréhensibles ;
  - soyez poli sans en faire des tonnes.
- Relisez-vous avant l'envoi.
- Demandez-vous si ce mail est indispensable. Le meilleur mail est celui qui n'est pas envoyé.

Mauvais exemple de mail

Objet : Urgent
Bonjour Monsieur le professeur, je ne comprends vraiment pas le cours numéro 3.
Le partiel est très bientôt et je vous promets avec d'autres étudiants on a essayé mais il y a trop de choses qu'on ne comprend pas est-ce que ce serait possible de ne pas poser de question sur ce cours au partiel s'il-vous-plaît. Merci d'avance.

Mauvais exemple de mail

Objet : Urgent
Bonjour Monsieur le professeur, je ne comprends vraiment pas le cours numéro 3.
Le partiel est très bientôt et je vous promets avec d'autres étudiants on a essayé mais il y a trop de choses qu'on ne comprend pas est-ce que ce serait possible de ne pas poser de question sur ce cours au partiel s'il-vous-plaît. Merci d'avance.

- Objet à la fois impoli (ne pas imposer son urgence à autrui) et dénué d'information.
- Pas de présentation, je ne sais pas de quel cours il s'agit.
- Trop imprécis sur les difficultés liées au cours, quelles notions ne sont pas comprises ?
- Demander des explications sur des points précis plutôt que de demander de ne pas évaluer quelque chose.
- « Merci d'avance » est impoli car veut forcer la main de son interlocuteur.

Bon exemple de mail

Objet : [Unix] Fichiers de configuration

Bonjour,

Je suis une de vos étudiantes en Unix.

Dans le transparent 9 du cours 3, vous évoquez les fichiers de configuration `.bashrc`, `.vimrc` et `.zshrc` de l'utilisateur.

J'ai regardé dans mon répertoire personnel avec la commande `ls -al` mais je n'ai que `.bashrc`, est-ce normal ?

Faut-il connaître les autres fichiers de configuration, et si oui, à quoi servent-ils ?

Cordialement,

`-`

Alice Bob, 12212345

Bon exemple de mail

Objet : [Unix] Fichiers de configuration

Bonjour,

Je suis une de vos étudiantes en Unix.

Dans le transparent 9 du cours 3, vous évoquez les fichiers de configuration `.bashrc`, `.vimrc` et `.zshrc` de l'utilisateur.

J'ai regardé dans mon répertoire personnel avec la commande `ls -al` mais je n'ai que `.bashrc`, est-ce normal ?

Faut-il connaître les autres fichiers de configuration, et si oui, à quoi servent-ils ?

Cordialement,

-

Alice Bob, 12212345

- L'objet apporte immédiatement du contexte, on sait tout de suite de quoi il est question.
- Les questions sont détaillées et concise, le contexte est bien donné.
- L'étudiante fait part de ses recherches et initiatives.
- Les questions sont séparées par un passage à la ligne, je peux répondre directement dans le texte à chacune d'entre elles.

Vos enseignants

Dans les matières scientifiques, ils sont généralement enseignants-chercheurs au LAGA ou au LIPN sauf Alain Rousseau et moi-même qui sommes enseignants à plein temps.

algèbre 1	Christian Ausoni, Alain Rousseau, Isabelle Vidal
analyse 1	Henry de Thélin, Alain Rousseau, Isabelle Vidal
programmation 1	Roland Grappe, Étienne André, Francesco Demelas, Ikram Garfatta
unix	Pierre Rousselin, Alexandre Louvet
preuves formelles	Micaela Mayero, Marie Kerjean, Pierre Rousselin

Réussir sa L1 DL

- Vu dans les transparents de la grande réunion des primo-arrivants : Assiduité + Travail $\implies$ réussite.

Réussir sa L1 DL

- Vu dans les transparents de la grande réunion des primo-arrivants : Assiduité + Travail $\Longrightarrow$ réussite.
- Ce sont certainement des conditions *nécessaires* mais pas suffisantes !
- Un étudiant qui est tout le temps là et travaille d'arrache-pied à apprendre par cœur sans comprendre ne peut pas réussir.

Réussir sa L1 DL

- Vu dans les transparents de la grande réunion des primo-arrivants : Assiduité + Travail $\Longrightarrow$ réussite.
- Ce sont certainement des conditions *nécessaires* mais pas suffisantes !
- Un étudiant qui est tout le temps là et travaille d'arrache-pied à apprendre par cœur sans comprendre ne peut pas réussir.
- Vous devez certes êtres présents et travailleurs mais surtout, en étant actifs.
- Les mathématiques et l'informatique sont des activités. On fait des mathématiques et on fait de l'informatique.

Exemple : « apprendre une démonstration en mathématiques »

- Ne pas apprendre mot à mot la démonstration : ça prend trop de temps et ne sert à rien.
- Commencer par essayer de la comprendre de fond en comble. Est-ce qu'il y a des points qui restent flous pour vous ? Des détails qui n'en sont pas ? Essayer (vraiment) de clarifier les choses.
- Il est normal de passer plusieurs heures à essayer de vraiment bien comprendre une démonstration. C'est une *activité*, vous aurez certainement à sortir du papier et un crayon pour combler les trous. Peut-être aurez-vous à dérouler la démonstration sur des exemples pour la comprendre.
- Identifier les points importants, les distinguer des détails.
- À la fin, vous devez être capable d'écrire vous-même cette démonstration avec votre propre niveau de détail, vos notations, etc.

Exemple : « comprendre un programme en C »

- Ne pas apprendre mot à mot le programme : ça prend trop de temps et ne sert à rien.
- Il est certainement bon de recopier (vraiment, pas de le copier-coller) le programme sur machine, ça rend la compréhension plus facile, vous fait rentrer la syntaxe dans les doigts, et vous permet ensuite de l'essayer (compiler et exécuter) et de le modifier.
- Le programme a peut-être plusieurs parties bien découpables, les identifier et comprendre leurs rôles.
- Certains passages peuvent être obscurs : faîtes de rapides traces sur papier.
- Identifier les choses nouvelles ou belles de ce programme pour pouvoir les réutiliser.
- À la fin vous devriez pouvoir écrire un programme qui a le même objectif, mais tout seuls et « à votre sauce ».

Informatique

- L'université met à votre disposition des salles de TP avec des machines sous Linux.
- Mais il est très très souhaitable que vous ayiez votre propre machine. Au pire, l'université peut vous en prêter.
- Vous aurez besoin d'un système Unix (et un compilateur C), donc, par exemple :
  - un Mac (mais c'est cher) ou
  - une machine sous Windows avec *dual boot* Linux ou
  - (moins bien mais plus simple) une machine Windows avec WSL (*Windows Subsystem for Linux*) ou bien une *virtual box* sous Linux.
  - Pire des cas : guacamole (ou mieux, **`ssh`**) sur les machines de l'université.
- Inutile d'avoir une machine puissante.
- Il faudrait vraiment régler ça maitenant, car vous avez du temps.

Le plus important pour votre réussite : l'entraide

- Traditionnellement, l'entraide est *très développée en licence maths/info et encore plus en DL.*
- Faîtes un serveur discord, ou autre !
- Soyez toujours aimables et soigneux dans vos interactions, soyez attentifs les uns des autres.
- Quand un étudiant explique quelque chose à un autre, c'est très positif *pour les deux étudiants.*
- Attention à la fausse aide : donner son travail à un autre pour qu'il le recopie (ou le copie-colle si c'est du code) est néfaste pour tout le monde.
- Normalement, l'ambiance en classe est à la fois très agréable et studieuse.
- N'hésitez pas à aller au tutorat : ce sont souvent d'anciens étudiants, passés par cette licence (maths ou info ou DL).

Quelques statistiques de l'année dernière

Sur 65 inscrits en DL1 l'année dernière,

- 5 défaillants au N1 (donc ne sont pas venus ou presque pas)
- 20 sont passés en parcours aménagé après le N1
- 2 changements d'université à l'issue du N1
- 10 redoublements à l'issue du N2
- 1 passage en L2 maths
- 27 admis en DL2 dont :
  - 10 mentions assez bien
  - 5 mentions bien
  - 4 mentions très bien
  - 2 félicitations du jury

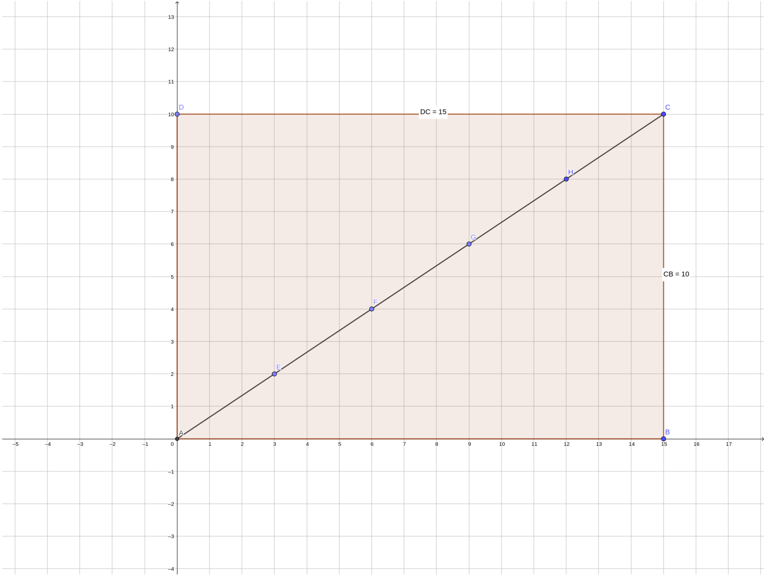
Où sommes-nous ?

Double-licence

Personnel de l'université

Réussir en DL1

Un problème Maths – Info


DC = 15
CB = 10
A
B
C
D
E
F
G
H

Problème

Problème

Soit une feuille quadrillée sur laquelle est dessiné un rectangle de largeur m et de longueur n en « utilisant le quadrillage. » Combien y a-t-il de points « du quadrillage » sur une diagonale de ce rectangle ?

Modélisation

- Bien qu'assez précis pour être compris par certaines personnes, l'énoncé précédent n'est pas un énoncé mathématique.
- Transformer un énoncé en langage naturel en énoncé mathématique est appelé *modéliser*.

Modélisation

- Bien qu'assez précis pour être compris par certaines personnes, l'énoncé précédent n'est pas un énoncé mathématique.
- Transformer un énoncé en langage naturel en énoncé mathématique est appelé *modéliser*.

Problème modélisé

Dans le plan muni d'un repère cartésien, on considère les points $O(0, 0)$ et $A(m, n)$ avec m et n entiers naturels non nuls.

Modélisation

- Bien qu'assez précis pour être compris par certaines personnes, l'énoncé précédent n'est pas un énoncé mathématique.
- Transformer un énoncé en langage naturel en énoncé mathématique est appelé *modéliser*.

Problème modélisé

Dans le plan muni d'un repère cartésien, on considère les points $O(0,0)$ et $A(m,n)$ avec m et n entiers naturels non nuls. Combien de points à coordonnées entières appartiennent au segment $[OA]$?

Analyse

On part d'une supposée solution quelconque et on cherche des *conditions nécessaires* sur les paramètres de cette solution.

- Soit $P(a, b)$ un point à coordonnées entières sur $[OA]$.
- Alors...

Analyse

On part d'une supposée solution quelconque et on cherche des *conditions nécessaires* sur les paramètres de cette solution.

- Soit $P(a, b)$ un point à coordonnées entières sur $[OA]$.
- Alors...
- (Théorème de Thalès, ou équation de droite, ou ...)
- $an = bm$, avec $0 \leq a \leq m$ et $0 \leq b \leq n$.

Exemples :

- avec $m = 15$ et $n = 12$, l'équation devient, après simplification par

Analyse

On part d'une supposée solution quelconque et on cherche des *conditions nécessaires* sur les paramètres de cette solution.

- Soit $P(a, b)$ un point à coordonnées entières sur $[OA]$.
- Alors...
- (Théorème de Thalès, ou équation de droite, ou ...)
- $an = bm$, avec $0 \leq a \leq m$ et $0 \leq b \leq n$.

Exemples :

- avec $m = 15$ et $n = 12$, l'équation devient, après simplification par 3, $4a = 5b$,

Analyse

On part d'une supposée solution quelconque et on cherche des *conditions nécessaires* sur les paramètres de cette solution.

- Soit $P(a, b)$ un point à coordonnées entières sur $[OA]$.
- Alors...
- (Théorème de Thalès, ou équation de droite, ou ...)
- $an = bm$, avec $0 \leq a \leq m$ et $0 \leq b \leq n$.

Exemples :

- avec $m = 15$ et $n = 12$, l'équation devient, après simplification par 3, $4a = 5b$, qu'en dire ?
- avec $m = 4$ et $n = 9$, ...

Analyse

On part d'une supposée solution quelconque et on cherche des *conditions nécessaires* sur les paramètres de cette solution.

- Soit $P(a, b)$ un point à coordonnées entières sur $[OA]$.
- Alors...
- (Théorème de Thalès, ou équation de droite, ou ...)
- $an = bm$, avec $0 \leq a \leq m$ et $0 \leq b \leq n$.

Exemples :

- avec $m = 15$ et $n = 12$, l'équation devient, après simplification par 3, $4a = 5b$, qu'en dire ?
- avec $m = 4$ et $n = 9$, ...on ne peut pas simplifier.
- avec $m = 10$ et $n = 10$?

Plus grand commun diviseur

Définition

Le plus grand diviseur commun de deux entiers naturels m et n non tous les deux nuls est le plus grand entier d qui divise à la fois m et n. On le note ici $m \wedge n$.

Plus grand commun diviseur

Définition

Le plus grand diviseur commun de deux entiers naturels m et n non tous les deux nuls est le plus grand entier d qui divise à la fois m et n. On le note ici $m \wedge n$.

Exemples :

- $15 \wedge 12 =$

Plus grand commun diviseur

Définition

Le plus grand diviseur commun de deux entiers naturels m et n non tous les deux nuls est le plus grand entier d qui divise à la fois m et n. On le note ici $m \wedge n$.

Exemples :

- $15 \wedge 12 = 3$.
- $4 \wedge 9 =$

Plus grand commun diviseur

Définition

Le plus grand diviseur commun de deux entiers naturels m et n non tous les deux nuls est le plus grand entier d qui divise à la fois m et n. On le note ici $m \wedge n$.

Exemples :

- $15 \wedge 12 = 3$.
- $4 \wedge 9 = 1$.
- $10 \wedge 10 =$

Plus grand commun diviseur

Définition

Le plus grand diviseur commun de deux entiers naturels m et n non tous les deux nuls est le plus grand entier d qui divise à la fois m et n. On le note ici $m \wedge n$.

Exemples :

- $15 \wedge 12 = 3$.
- $4 \wedge 9 = 1$.
- $10 \wedge 10 = 10$.

Comment calculer le PGCD ?

- Méthode pour pouvoir calculer le PGCD de deux nombres ?
- Un problème d'informatique !
- Commençons par une méthode « naïve », c'est-à-dire la première qui vient à l'esprit.

Comment calculer le PGCD ?

- Méthode pour pouvoir calculer le PGCD de deux nombres ?
- Un problème d'informatique !
- Commençons par une méthode « naïve », c'est-à-dire la première qui vient à l'esprit.
- Essayer avec tous les entiers plus petits que m et n et ne garder que le plus grand qui les divise tous les deux.

Comment calculer le PGCD ?

- Méthode pour pouvoir calculer le PGCD de deux nombres ?
- Un problème d'informatique !
- Commençons par une méthode « naïve », c'est-à-dire la première qui vient à l'esprit.
- Essayer avec tous les entiers plus petits que m et n et ne garder que le plus grand qui les divise tous les deux.

Algorithme 1 : PGCD de deux entiers (très naïf)

Données : Deux entiers naturels m et n
Résultat : Valeur de $m \wedge n$
Variables : *entiers naturels* d *et* r
$d \longleftarrow 1$
$r \longleftarrow 0$
tant que $d \leq \min(m, n)$ **faire**
  si d *divise à la fois* m *et* n **alors**
    $r \longleftarrow d$
  $d \longleftarrow d + 1$
retourner *le contenu de* r

De l'algorithme au programme

- L'algorithme contient encore des imprécisions, il n'est pas compréhensible par une machine.
- On va le programmer en langage C. Ceci est un exemple, ce n'est pas obligé de tout comprendre aujourd'hui !
- Un autre programme appelé « compilateur C » va se charger de transformer notre programme en une suite d'instructions compréhensibles par la machine.

Contenu du fichier pgcd_naif.c

```
#include <stdio.h>
#include <stdlib.h>
int main(int argc, char *argv[])
{
        unsigned m, n, d, min, gcd;
        m = atoi(argv[1]);
        n = atoi(argv[2]);
        if (m < n)
                min = m;
        else
                min = n;
        d = 1;
        gcd = 0;

        while (d <= min) {
                if (m % d == 0 && n % d == 0)
                        gcd = d;
                d = d + 1;
        }
        printf("pgcd(%u, %u) = %u\n", m, n, gcd);
        return 0;
}
```

Compilation et exécution

- Compiler le programme en C et appeler `pgcd1` l'exécutable obtenu :

```
$ gcc -Wall pgcd_naif.c -o pgcd1
$
```

- Le compilateur n'indique ni erreur ni avertissement (c'est rare), on peut exécuter.
- On exécute avec comme arguments 15 et 12 :

```
$ ./pgcd1 15 12
pgcd(15, 12) = 3
```

- Idem mais avec des grands nombres et en chronométrant :

```
$ time ./pgcd1 456789123 741852963
pgcd(456789123, 741852963) = 9

real        0m2,283s
user        0m2,265s
sys         0m0,004s
```

- ... et encore d'autres tests et éventuellement des améliorations.

Algorithme d'Euclide

- Maintenant on *sait calculer* le PGCD de deux entiers naturels.
- C'est déjà une grande victoire : le PGCD de deux entiers n et m est *calculable*[2] avec de l'ordre de $\min(m, n)$ divisions euclidiennes.
- Mais Euclide (env. -300) savait déjà faire mieux...
- Un des plus anciens algorithmes, toujours utilisé en pratique (on n'a guère fait mieux).

2. Il y a en effet des choses qui ne sont pas calculables.

Algorithme d'Euclide

Algorithme 2 : Algorithme d'Euclide

Données : Deux entiers naturels a et b
Résultat : Valeur de $a \wedge b$
Variables : *entier* r
tant que $b > 0$ **faire**
  $r \longleftarrow$ le reste de la division de a par b
  $a \longleftarrow b$
  $b \longleftarrow r$
retourner *le contenu de* a

Algorithme d'Euclide

Algorithme 2 : Algorithme d'Euclide

Données : Deux entiers naturels a et b
Résultat : Valeur de $a \wedge b$
Variables : *entier* r
tant que $b > 0$ **faire**
  $r \longleftarrow$ le reste de la division de a par b
  $a \longleftarrow b$
  $b \longleftarrow r$
retourner *le contenu de* a

Exemple : $a = 84$, $b = 60$

1. $84 = 60 \times 1 + 24$
2. $60 = 24 \times 2 + 12$
3. $24 = 12 \times 2 + 0$

Donc $84 \wedge 60 = 12$.

Fichier pgcd_euclide.c

```c
#include <stdio.h>
#include <stdlib.h>

int main(int argc, char *argv[])
{
        unsigned m, n, a, b, r;
        m = atoi(argv[1]);
        n = atoi(argv[2]);

        a = m;
        b = n;

        while (b > 0) {
                r = a % b;
                a = b;
                b = r;
        }
        printf("pgcd(%u, %u) = %u\n", m, n, a);
        return 0;
}
```

Compilation et exécution

```
$ gcc -Wall pgcd_euclide.c -o pgcd2
$ ./pgcd2 15 12
pgcd(15, 12) = 3
$ ./pgcd2 60 48
pgcd(60, 48) = 12
$ ./pgcd2 101 2017
pgcd(101, 2017) = 1
$ time ./pgcd2 456789123 741852963
pgcd(456789123, 741852963) = 9

real        0m0,002s
user        0m0,001s
sys         0m0,001s
```

Compilation et exécution

```
$ gcc -Wall pgcd_euclide.c -o pgcd2
$ ./pgcd2 15 12
pgcd(15, 12) = 3
$ ./pgcd2 60 48
pgcd(60, 48) = 12
$ ./pgcd2 101 2017
pgcd(101, 2017) = 1
$ time ./pgcd2 456789123 741852963
pgcd(456789123, 741852963) = 9

real        0m0,002s
user        0m0,001s
sys         0m0,001s
```

En fait on peut montrer que le nombre de divisions euclidiennes dans l'algorithme d'Euclide avec pour entrées m et n est majoré par $C \log(\min(m, n))$ avec $C > 0$ constant, contre $\min(m, n)$ pour l'algorithme « naïf ».

Compilation et exécution

```
$ gcc -Wall pgcd_euclide.c -o pgcd2
$ ./pgcd2 15 12
pgcd(15, 12) = 3
$ ./pgcd2 60 48
pgcd(60, 48) = 12
$ ./pgcd2 101 2017
pgcd(101, 2017) = 1
$ time ./pgcd2 456789123 741852963
pgcd(456789123, 741852963) = 9

real        0m0,002s
user        0m0,001s
sys         0m0,001s
```

En fait on peut montrer que le nombre de divisions euclidiennes dans l'algorithme d'Euclide avec pour entrées m et n est majoré par $C\log(\min(m,n))$ avec $C > 0$ constant, contre $\min(m,n)$ pour l'algorithme « naïf ». Il y a un monde entre les deux !

À faire ou à essayer :

- Récupérer ces transparents sur moodle dans « Licence Service Pédagogique 22-23 ».
- Une fois équipé d'un compilateur C (sous Linux `gcc`, sous Mac `clang`) et d'un éditeur de texte, recopier les programmes, les compiler et les exécuter. Essayer de comprendre certaines parties en faisant des modifications.
- Conjecturer et prouver une formule sur le nombre de points à coordonnées entières dans le problème de mathématiques. Si vous avez suivi l'option maths expertes, vous pouvez essayer de la démontrer. Indice : après simplification par $n \wedge m$, utiliser le lemme de Gauss :
  `https://fr.wikipedia.org/wiki/Lemme_d%27Euclide`
- Soient q et r le quotient et le reste de la division euclidienne d'un entier naturel non nul a par un entier naturel b. On suppose que $a \geq b$. Prouver que $r < \frac{a}{2}$.
- En déduire que dans ce cas, le nombre d'itérations dans l'algorithme d'Euclide pour calculer le PGCD de a et b est majoré par $\log_2(a)$, où $\log_2$ est le logarithme en base 2. Et si $a < b$?