Calculabilité TD : Lucas.isenmann@umontpellier.fr.

[a|x] valeur retournée
↑ algo ↑ entrée

Diverge : Ne s'arrête pas Converge : s'arrête.

Domaine de convergence : $W_a = \{x : [a|x]\downarrow\}$

Totale $\Leftrightarrow W_a = \mathbb{N}$.

$Im(a) = \{[a|x], x \in \mathbb{N}\}$

TD1

Exercice 1 :

E est énumérable, donc il existe un programme (fonction) qui énumère tous les éléments de l'ensemble.

$E = \{0, 1, 2\}$

```
def a(x)
  if x < 3 :
    return 1
  else
    while true :
      continue
```

2) E admet une fonction d'énumération calculable. $Im(b) = E$?

```
def b(x)
  if x < 3
    return x
  while true:
    continue
```

$E = \{0, 1, 4, 9, 16, \dots\}$
$= \{n^2 : n \in \mathbb{N}\}$

```
def b(x)
  return x*x
```

$Im(b) : E$

```
def a(x) :
  n = int( sqrt(x))
  if n×n = x
    return 0
  else : ↑
```

1→3 step <b, x, t> = 0 veut dire que l'algo n'a pas pour l'instant rien retourné au bout de t étapes

On veut construire b tel que $Im(b) = E$ et $W_b = \mathbb{N}$ $\boxed{\text{totale}}$ $E \neq \emptyset$

```
def b(x) :
  si [a|x]↓
    return x
  else
    while true
```

Correction :

On suppose $E = W_a$ et $e \in E$ car E est non vide. On peut alors écrire la fonction totale.

$b : \langle x, t \rangle \mapsto$ if step $\langle a, x, t \rangle \neq 0$
  | return x
else
  | return e

On veut alors prouver que l'image de cette fonction b est E ($Im[b|\cdot] = E$)

- Montrons d'abord que $E \subset Im[b|\cdot]$
  Par tout x de E, il existe un temps t tel que step $\langle a, x, t \rangle \neq 0$. Alors $[b|\langle x, t\rangle] = x$ donc x est dans $Im[b|\cdot]$.
- Montrons ensuite que $Im[b|\cdot] \subset E$
  Par tout x de $Im[b|\cdot]$ alors
  - $x = a_e$, et $a_e \in E$ donc $x \in E$
  - $x \neq a_e$, alors il existe t tel que step $\langle a, x, t \rangle \neq 0$, donc $x \in E$ (W_a).

Donc $E = Im([b|\cdot])$

2) On veut montrer que 3=>2.

- Si E est vide, b est un programme qui plante tout le temps, son image est donc l'ensemble vide.
- Sinon E admet une fonction b d'énumération totale. b est donc une fonction d'énumération.

3) $a : \langle x \rangle \mapsto \langle y, t \rangle \leftarrow 0$

```
while step <b, y, t> ≠ x+1
    <y,t>++
return 1
```

TD 2:

Exercice 3: Rappel : $\Leftrightarrow$ énumérable $W_a = E$, fonction énumérable totale $Im(b) = E$.

2) $A \cap B$ énumérables ? Soient A et B deux algos tels que $W_a = A$
$W_b = B$

```
fonction c(x)
    a(x)
    b(x)
    return 0
fin fonction
```

si il retourne 0, c'est bon si il n'est pas dans $A \cap B$, il va planter sur a ou b.

$\boxed{W_c = A \cap B \quad , \quad c(x)\downarrow \Leftrightarrow x \in A \cap B}$

2ème preuve : (Avec image)

Soient e et f des algos, tel que $Im(e) = A$ $(\forall a \in A, \exists n \in \mathbb{N} : e(n) = a)$
$Im(f) = B$

Soit $g(x)$ une fonction que l'on construit,

```
g(n)
    while (f(x) ≠ e(n))
        x++
    return e(n) (ou f(x))
```

	0	1	2	3	4
A =	0	2	4	6	8
B :	0	1	4	9	16

$e(3) = 6$ mais $6 \notin B$

1) $A \cup B$ énumérable ? Soient a, b des algos tels que $A = W_a$
$B = W_b$

```
fonction c(x):
    t=0
    while (step(x,t,a) = 0 et step(b,x,t) = 0)
        t ← t+1
    retourne 0.
```

Il faut lancer a et b en même temps car si l'élément $\notin$, alors boucle à l'infini.

"a(x) ou b(x) a convergé après t étapes" = step (a,x,t) ≠ 0 ou step(b,x,t) ≠ 0

2ème preuve : Soient e et f tels que $Im(e) = A$ et $Im(f) = B$

```
fonction g(n)
    si n % 2 = 0
        retourne e(n/2)
    sinon
        retourne f((n-1)/2)
```

va faire tous les indices.

A : 0 2 4 6

B : 0 1 4 9

$A \cup B$: 0 0 1 2 4 6 4 9

Pairs

carré

Un sur deux est le carré.

$\boxed{Im(g) = A \cup B}$

- $Im(g) \subseteq A \cup B$ Soit $x \in Im(g)$. Donc $\exists n \in \mathbb{N}$ tel que $g(n) = x$

Si n est pair $x = g(n) = e(n/2) \in A$

Si n est impair $x = g(n) = f((n-1)/2) \in B$

Donc $x \in A \cup B$, donc $Im(g) \subset A \cup B$

$\boxed{A \cup B \subset Im(g)}$ .....

Exercice 2 :

$g(n) = [x|x]' + 1$ est calculable ?

On peut numéroter les (algos / fonctions) de ce système, le $x$ième algo sur l'entrée x s'appelle $[x|.]'$

L'apostrophe représente un autre système de calcul.

Le point $\bullet$ représente n'importe quelle entrée.

Montrer qu'elle est calculable.

```
def g(x)
    return x(x)+1
```

$g = comp(succ, x)$

2) $\exists n$ tq $g : a_n = n$

$g(n) = [n|n]' + 1$

$= \underset{g}{[n|n]'}$

Conclusion : Un tel système n'existe pas.

Exercice 1 :

1) E énumérable infini, alors fonction d'énumération totale bijective.

```
fonction f(n)
    S <- {}          n=1
    i <- 0
    tant que taille(S) < n
        | si e(i) ∉ S :
        |     ajouter(S, e(i))
        | i <- i+1
    tant que e(i) ∈ S
        | i <- i+1
    return e(i)
```

	0	1	2	3	4	5	6
E=	4	6	3	3	0	6	7

Non bijectif.

	0	1	2	3	4	5
E=	4	6	3	0	7	

Virer les doublons.

2) E récursif ssi admet fonction d'énum croissante

E récursif : $\exists a$ tq $x \in E \Leftrightarrow a(x)$ Vrai

$\Leftarrow$ On suppose e énumération totale croissante

```
fonction a(x)
    | i <- 0
    | while e(i) < x
    |     | i <- i+1
    | si e(i) = x
    |     return V
    | return F
```

	0	1	2	3	4	5
E =	0	2	2	3	6	10

TD3 :

Pas $\emptyset$ et pas $\mathbb{N}$

Rappel : Soit P prédicat tq ... et qu'il soit non trivial ($P \neq \emptyset$ et $P \neq \mathbb{N}$) avec $a(x) = b(x)\ \forall x \Rightarrow P(a) = P(b)$

(Un prédicat est soit une fonction $P : \mathbb{N} \to$ bool $\{0,1\}$
ou $P \subseteq \mathbb{N}$)

$\Rightarrow$ P est non récursif ou non décidable.

Un prédicat est décidable si on peut dire si oui ou non l'élément est dans E.

Preuve :

Par l'absurde, on suppose $\exists C_p$ un algo tel que $C_p(a) =$ Vrai $\Leftrightarrow P(a)$ Vrai.

On va créer une fonction $f(x)$

```
def f(x)
    if Cp(f)
        | return b(x)
    else
        | return a(x)
```

Si $C_p(f)$ Vrai

alors $f(x) = b(x)\ \forall x$

$\Rightarrow f = b$? Non pas forcément.

$\Rightarrow \underset{\text{Vrai}}{P(f)} = \underset{\text{Faux}}{P(b)}$ Contradiction

Paraît si $Cp(f)$ Faux. Donc indécidable.

```
def G(x)
    return 0;
```

vérifie pas p.
P(b) = Faux

```
def c(x)
    ⋮
    return 0;
```

P(A) = Vrai

Exercice 1 : Soit $A = \{x, \forall y [x|y]\downarrow\}$ Un programme qui converge tout le temps $\forall y$ en entrée.

1)

```
def a(x):
    return 0;
```

$a \in A$ $\xleftarrow{}$ converge $\forall y$

```
def b(x)
    while True
```

$b \notin A$

$\longrightarrow$ Non trivial, donc non récursif. (Avec le théorème de Rice).

Soient f et g tel que $f(x) = g(x)$ $\forall x$ algos. $\Rightarrow P(f) = P(g)$. A est sémantique.

$A = \{x, x \in C$. C est totale.

2) $K < A$. K est l'ensemble des algos tel que $K = \{x, [x|x]\downarrow\}$ Ensembles des algos qui calculé sur son nombre converge.

```
def 1(x)
    return 0;
```

$[1|1]\downarrow$
donc $1 \in K$.

$A < B$ où A et B soit $\exists f$ un algo total, $x \in A \Leftrightarrow f(x) \in B$.

```
def a(<x,z>)
    si [x|x]↓
        return z
```

=

```
def a(u)
    x = π1(u)
    y = π2(u)
    si ___
```

$f(x) = a(<x,\cdot>) \mid f(x)$ est une fonction

$f(x)(z) = a(<x,z>)$ d'une variable z.

Montrons que : $x \in K \Leftrightarrow f(x) \in A$

Soit $x \in K$ ($[x|x]\downarrow$)

1) Montrons que $f(x) \in A$ montrer que $f(x)(z)\downarrow$ $\forall z$

$f(x)(z) = z$ $\forall z$ donc converge $\forall z$ donc $f(x) \in A$.

2) Soit $x \notin K$, Mq $f(x) \notin A$

Comme $x \notin K$, $[x|x]\uparrow$

Donc $f(x)(x)\uparrow$. Donc $f(x) \notin A$.

3) $K < \bar{A}$

```
def b(<x,z>)
    si step(x, x, z) = 0
        return z.
    else
        ⊥
```

$f(x) = b(<x,\cdot>) \mid f(x)$ est une fonction

$f(x)(z) = b(<x,z>)$ d'une variable z.

Mq $x \in K \Leftrightarrow f(x) \in \bar{A}$

1) Soit $x \in K$ donc $[x|x]\downarrow$. Mq $f(x) \in \bar{A} \Leftrightarrow \exists z$ tq $f(x)(z)\uparrow$??

$\exists t$ tq step $(x, x, t) \neq 0$.

$f(x)(t)\uparrow$ ✓ donc $f(x) \in \bar{A}$

2) Soit $x \notin K$

Mq $f(x) \notin \bar{A} \Leftrightarrow f(x) \in A$. Soit z, $f(x)(z)$

Comme $x \notin K$, $[x|x]\uparrow$, donc step $(x, x, t) = 0$ $\forall t$.

$f(x)(z) = z\downarrow$ Donc $f(x) \in A$. Conclusion $x \in K \Leftrightarrow f(x) \in \bar{A}$

4) Th de Post : si A énumérable et $\bar{A}$ aussi alors A récursif.

Par l'absurde, si ce n'est pas vrai. Donc soit A est énumérable soit $\bar{A}$ l'est.

```
def a(<x, z>):
    if [x | x]↓
        —
```

Squelette.

Supposons que A soit énumérable.

Q3) $K < \bar{A} \Rightarrow \bar{K} < A$. (Thm 3)

$\Rightarrow \bar{K}$ énumérable (Thm 2)

Thm 2 : $A < B$ et B énumérable $\Rightarrow A$ est énumérable.

Thm 3 : $A < B \Rightarrow \bar{A} < \bar{B}$.

Q2) $K < A \Rightarrow K$ énumérable (Thm 2)

Donc K et $\bar{K}$ énumérable. Donc K est récursif

Absurde donc A pas énumérable

Si $\bar{A}$ énumérable, c'est pareil. Dans les deux cas on a une contradiction. Donc ni A ni $\bar{A}$ est énumérable.

Exercice 2

1) $B : \{x : [x|0]\downarrow \text{ et } [x|1]\uparrow\}$

```
def a(x)
    if x ≠ 1
        return 0
    else
        while True
```

```
def b(x)
    return 0
```

$b \notin B$ (car $[b|1]\downarrow$) (c'est semantique)

Rice $\Rightarrow B$ pas récursif.

$B = \{x, x \in C\}$ C converge en 0 et diverge en 1.

TD4 :

Exercice 1 :

1)
```
def ex(n):
    return 0
```

$ex \in C$ return 0 à chaque fois donc $ex(0) = ex(1)$

```
def cex(n)
    return n
```

$cex \notin C$ C n'est pas décidable. $cex(0) \neq cex(1)$

```
def a(x)
    if [x|0] = [x|1]
        return 0.
    else
        while True
```

$C = Dom(a) \Rightarrow$ Enumérable.

2)
```
def ex(n)
    return n
```

$ex \in D$

non trivial

```
def cex(n)
    return 0
```

$cex \notin D$

Rice $\Rightarrow$ Indécidable.

$\boxed{K < D}$ f tq $x \in K \Leftrightarrow f(x) \in D$ f calculable totale $\boxed{K < D}$

$K = \{x : [x|x]\downarrow\}$

```
def a(<x, z>):
    if [x | x]↓
        return z
```

$x \in K \Rightarrow f(x) \in D$? 1

Soit $x \in K$

$[f(x)](z)\downarrow$ et $[f(x)|z] = z$

$g(z) : z \mapsto f(x)(z)$

Donc $g \in D$

$x \notin K \Rightarrow f(x) \notin D$ 2

Soit $x \notin K$

Donc $[x|x]\uparrow$

$[f(x)|z]\uparrow$

$g : z \mapsto f(x)(z)$

$[g|0]\uparrow$ donc g pas totale

donc $g \notin D$

```
def a(<x, z>):
    if [x | x]↓
        —
```

Squelette.

$f(x) = a(x, \cdot) = S^1_1(a, x)$

Méthode : • $K < D$ et $K < \bar{D}$ pour montrer que c'est pas énumérable.

• Trouver a tel que $D = W_a$ $\Rightarrow$ énumérable. squelette réponse

```
def a(<x, z>):
    if [x|x]↓
        —
```

Squelette.

```
def b(<x,z>):
    if step(x,x,z)=0
        —
    else
        —
```

$\boxed{K < \bar{D}}$ f tq $x \in K \Leftrightarrow f(x) \in \bar{D}$

$x \notin K \Rightarrow f(x) \notin \bar{D}$

Soit $x \notin K$ ($[x|x]\uparrow$)

$step(x,x,z) = 0 \quad \forall z$

$[f(x)|z] = z \; \forall z$

Donc $f(x) \in D$ et $f(x) \notin \bar{D}$

$x \in K \Rightarrow f(x) \in \bar{D}$

Soit $x \in K$ ($[x|x]\downarrow$)

$\exists t$ tq $step(x,x,t) \neq 0 \quad t > 0$

$f(x)$ n'est pas injective

$f(x)(0) = 0$

$f(x)(t) = 0$ $\Rightarrow f(x)$ pas injective

```
def b(<x,z>):
    if step(x,x,z)=0
        return z
    else
        while True
```

Indécidable : Rice, un algo dedans et un pas dedans

6) $H = \{premiers\}$

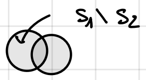
décidable : on trouve un algo qui dit si un entier est premier

```
def c(n)
    if n <= 1 : return False.
    k = 2
    while k<n
        if n%k==0;
            return False
        k += 1
    return True.
```

Décidable $\Rightarrow$ énumérable.

3) 4) 5) : Indécidables, et le 3 est énumérable. 4,5 méthode $K<D$ $K<\bar{D}$

Exercice 2 :



1) S_1 et S_2 pas $S_1 \setminus S_2$ énumérable

$S_1 = K \cup$ pairs

$S_2 = \{n : \text{divisible par } 4\} \cup K$

où K est arbitrairement choisi $\subseteq$ impairs.

$S_1 \setminus S_2 = \{n : \text{pairs pas divisible par } 4\}$ Pas de crochet = décidable.

numérotation des algorithmes.

```
def a(n):
    return n% ==0 and n%4≠0
```

$\Rightarrow$ décidable $\Rightarrow$ énumérables

Th