

# LSINC 1101

## Informatique 1 Introduction à la programmation

Modalités du cours



# Les titulaires

Pr. Charles Pecheur

- [Charles.Pecheur@uclouvain.be](mailto:Charles.Pecheur@uclouvain.be)
- Première partie du cours

Pr. Cristel Pelsser

- [Cristel.Pelsser@uclouvain.be](mailto:Cristel.Pelsser@uclouvain.be)
- Deuxième partie du cours

Pr. Kim Mens

- [Kim.Mens@uclouvain.be](mailto:Kim.Mens@uclouvain.be)
- Troisième partie du cours



A neon sign on a brick wall. The sign consists of two lines of text: 'QUIZ' on top and 'TIME' on the bottom. The letters are outlined in a vibrant pink color. The sign is enclosed in a rectangular frame made of neon tubes. The top and bottom horizontal bars of the frame are pink, while the left and right vertical bars are cyan. The background is a dark brick wall with a subtle purple and blue gradient.

QUIZ  
TIME

# Objectifs du cours d'informatique

A l'issue de ce cours, chaque étudiant sera en mesure de :

- **utiliser des outils** informatiques
- **utiliser** à bon escient les éléments d'un **langage de programmation** tel que Python,
- **concevoir et réaliser des programmes** corrects
  - de complexité moyenne
  - avec méthode et rigueur
- démontrer une bonne compréhension des concepts et de la méthodologie de la **programmation**
- et de la programmation **orientée-objet**
- utiliser la programmation pour le **projet de quadrimestre**

# Comment atteindre ces objectifs ?

- Apprentissage actif
  - Typiquement par **groupes** de 2
  - A partir de **problèmes concrets** (1 par semaine)
- Permet d'acquérir
  - le **savoir** : par l'**étude** et la **pratique**
  - le **savoir-faire** : par la **pratique** uniquement

# Organisation d'une semaine du cours

## Une mission par semaine

Jeudi			
	<i>cours magistral</i>	<b>Introduction</b>	Introduction à la matière liée au nouveau problème
Vendredi-weekend	<i>auto-apprentissage</i>	<b>Introduction</b>	Lecture du syllabus Faire les exercices préparatoires
Lundi	<i>séance tutorée</i>	<b>Bilan intermédiaire</b>	Réponses aux questions
Mercredi	<i>Travail en binôme</i>	<b>Réalisation</b>	Ecriture d'un programme Soumission au tuteur

# Organisation d'une semaine du cours

## Une mission par semaine

Jeudi	<i>cours magistral</i>	<b>Restructuration</b>	<i>Restructuration de la matière vue durant la semaine</i>
	<i>séance tutorée</i>	<b>Bilan final</b>	Feedback sur la mission
			Question de bilan final
<i>cours magistral</i>	<b>Introduction</b>	Introduction à la matière liée au nouveau problème	
Vendredi-weekend	<i>auto-apprentissage</i>	<b>Introduction</b>	Lecture du syllabus Faire les exercices préparatoires
Lundi	<i>séance tutorée</i>	<b>Bilan intermédiaire</b>	Réponses aux questions
Mercredi	<i>Travail en binôme</i>	<b>Réalisation</b>	Ecriture d'un programme Soumission au tuteur

# Syllabus du cours

Disponible en ligne

<https://syllabus-interactif.info.ucl.ac.be/index/info1-theory>

En anglais



Version adaptée du livre open source :

**How to Think Like a Computer Scientist –  
Learning with Python 3 (RLE)**

by Peter Wentworth, Jeffrey Elkner,  
Allen B. Downey, and Chris Meyers

open-source, disponible gratuitement



# A propos du syllabus

A **maîtriser** (et non mémoriser)

= **étudier** intelligemment

Lecture **obligatoire** !

Les références du cours s'y rapportent

Complémentaire aux transparents/capsules du cours

Disponible (en ligne) à l'examen (sur ordinateur)

(les exercices ne seront pas disponible en ligne à l'examen)



# Manuel d'exercices

Disponible en ligne

<https://syllabus-interactif.info.ucl.ac.be/index/info1-exercises>

En français

Enoncés de toutes les missions

- matière à lire
- exercices de démarrage (interactifs)
- réalisation de la mission
- exercices complémentaires



# Horaire LSINC1101

	Lu	Ma	Me	Je	Ve
8:00					
10:00					
10:00					
12:00					
13:30	APP			13:45 ! APP	
15:30				15:45 Cours	
16:00			remise travail 16:00		

# La matière

*La matière faisant l'objet de l'examen comprend tout ce qui a été dit ou montré au cours oralement, sur écran ou à l'aide d'autres media, et ne se limite donc exclusivement au texte du "syllabus du cours"*

Cours =  
cours magistral +  
séances tutorées



# Evaluation

*Un travail de programmation est effectué chaque semaine.*

*Une interrogation intermédiaire a lieu en milieu de quadrimestre, un **examen** en fin quadrimestre.*

*La **note finale** du cours prend en compte l'interrogation intermédiaire et les travaux durant le quadrimestre, en plus de la note de l'examen.*



# Evaluation

Interrogation à mi-quadrimestre (S8)

- écrit sur papier

Examen en janvier / juin / août

- sur ordinateur (INGInious)

Note =

$1/3$  Interro +  $2/3$  Examen

si Interro > Examen

$3/3$  Examen

si Examen  $\geq$  Interro

+ bonus éventuel

pour participation

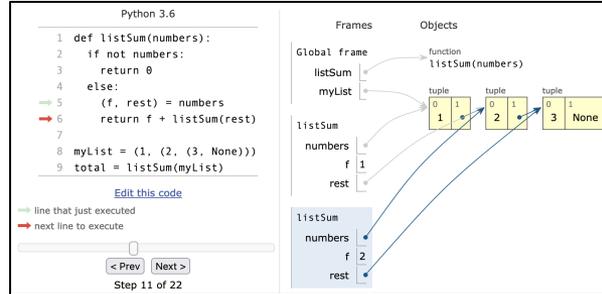
active aux missions

```

decompte.py x
1 i = 5
2 while i >= 0:
3     print(i)
4     i = i - 1
5 print("Decollage")

Shell x
Python 3.7.9 (bundled)
>>>
Python 3.7.9 (bundled)
>>> %Run decompte.py
5
4
3
2
1
0
Decollage
>>>

```



MoodleUCLouvain - Mes cours - Besoin d'aide? - Français (FR) - Mens Kim Etudiant

LSINC1101 – Introduction à la programmation (Charleroi)

Tableau de bord / Mes cours / LSINC1101

- Announces
- Forum d'aide
- Horaire Cours LSINC1101 Charleroi Q1 2021-2022: 74.6Ko Document PDF Modifié 11 sep 21, 17:02
- Enquête initiale

Vous êtes un nouvel étudiant? Merci de répondre à l'enquête initiale concernant votre familiarité avec l'informatique et votre parcours scolaire. Cela ne prend que quelques minutes.

Cette enquête est réalisée à chaque rentrée depuis 2012 à Louvain-la-Neuve et depuis 2020 à Charleroi.

Ressources

- Syllabus théorie
- Le syllabus du cours.
- Manuel d'exercices

## Info - Theory - Lists

Source: this section is heavily based on Chapter 11 of [ThinkCS].

A list is an ordered collection of values. The values that make up a list are called its **elements**, or its **items**. We will use the term *element* or *item* to mean the same thing. Lists are similar to strings, which are ordered collections of characters, except that the elements of a list can be of any type. Lists and strings --- and other collections that maintain the order of their items --- are called **sequences**.

### List values

There are several ways to create a new list; the simplest is to enclose the elements in square brackets ( [ and ] ):

```
ps = [10, 20, 30, 40]
qs = ["spam", "bungee", "swallow"]
```

The first example is a list of four integers. The second is a list of three strings. The elements of a list don't have to be the same type. The following list contains a string, a float, an integer, and (amazingly) another list:

```
zs = ["hello", 2.0, 5, [10, 20]]
```

A list within another list is said to be **nested**.

Finally, a list with no elements is called an **empty list**, and is denoted [ ].

INGenius - [INFO1] Informatique 1 - Introductio... - Session 2: Somme à compléter - Liste des cours - Kim Mens

### Session 2: Somme à compléter

compléter

Complétez le fragment de code suivant:

# Place dans sum la somme des n premiers entiers pairs

```
n = some_value
sum = 0
for ___ in _____:
    _____
```

Implémentation

```
1 n = some_value
2 sum = 0
3 for ___ in _____:
4     _____
```

Administration

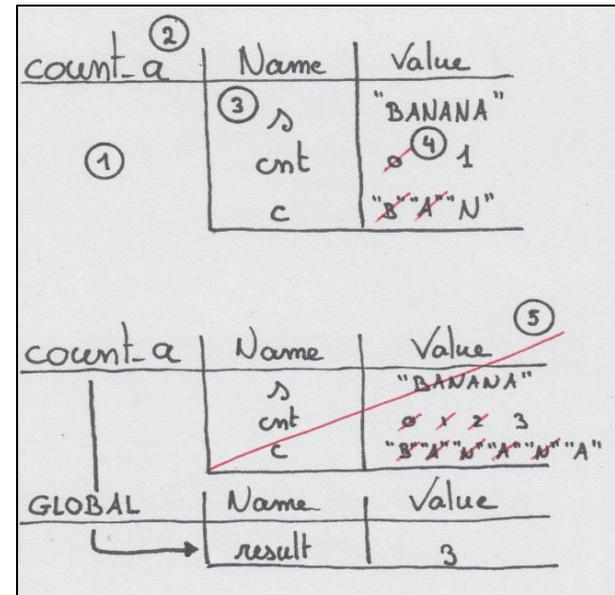
- Voir les soumissions
- Editer l'exercice
- Informations de débogage

Pour évaluation

- Meilleure soumission
- Pas de soumission

Soumettre

# Outils informatiques



# Thonny



The screenshot displays the Thonny Python IDE interface. At the top, there is a toolbar with icons for file operations, running, and stopping. Below the toolbar, a window titled 'decompte.py' contains the following Python code:

```
1 i = 5
2 while i >= 0:
3     print(i)
4     i = i - 1
5 print("Decollage")
```

Below the code editor, a 'Shell' window shows the execution of the script. It displays the Python version 'Python 3.7.9 (bundled)' and the prompt '>>>'. The user has entered the command '%Run decompte.py', and the output shows the numbers 5, 4, 3, 2, 1, 0, followed by the string 'Decollage'. The shell prompt '>>>' is visible at the bottom.

<https://thonny.org/>

# PythonTutor

Python 3.6

```
1 def listSum(numbers):  
2     if not numbers:  
3         return 0  
4     else:  
→ 5         (f, rest) = numbers  
→ 6         return f + listSum(rest)  
7  
8 myList = (1, (2, (3, None)))  
9 total = listSum(myList)
```

[Edit this code](#)

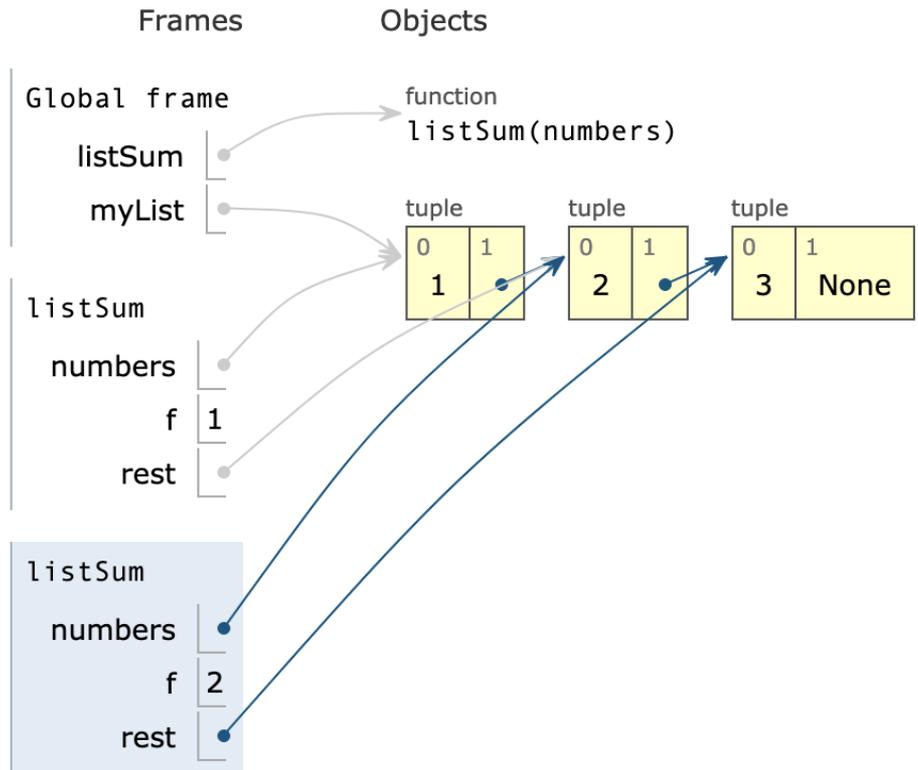
→ line that just executed

→ next line to execute



< Prev Next >

Step 11 of 22



<https://pythontutor.com/>

## LSINC1101 – Introduction à la programmation (Charleroi)

Tableau de bord / Mes cours / LSINC1101

 Annonces

 Forum d'aide

 Horaire Cours LSINC1101 Charleroi Q1 2021-2022 74.6Ko Document PDF Modifié 11 sep 21, 17:02

 Enquête initiale

**Vous êtes un nouvel étudiant ? Merci de répondre à l'enquête initiale concernant votre familiarité avec l'informatique et votre parcours scolaire. Cela ne prend que quelques minutes.**

Cette enquête est réalisée à chaque rentrée depuis 2012 à Louvain-la-Neuve et depuis 2020 à Charleroi.

### Ressources

 Syllabus théorie

Le syllabus du cours.

 Manuel d'exercices

# Moodle

# Lists

Source: this section is heavily based on Chapter 11 of [ThinkCS].

A **list** is an ordered collection of values. The values that make up a list are called its **elements**, or its **items**. We will use the term *element* or *item* to mean the same thing. Lists are similar to strings, which are ordered collections of characters, except that the elements of a list can be of any type. Lists and strings --- and other collections that maintain the order of their items --- are called **sequences**.

## List values

There are several ways to create a new list; the simplest is to enclose the elements in square brackets ( [ and ] ):

```
ps = [10, 20, 30, 40]
qs = ["spam", "bungee", "swallow"]
```

The first example is a list of four integers. The second is a list of three strings. The elements of a list don't have to be the same type. The following list contains a string, a float, an integer, and (amazingly) another list:

```
zs = ["hello", 2.0, 5, [10, 20]]
```

A list within another list is said to be **nested**.

Finally, a list with no elements is called an empty list, and is denoted [].

Syllabus  
en ligne :  
théorie

Accès à Inginious

Info1 - Exercices ▾

Notions de base ▾

Mission 1 - Mise en route ▾

Phase de démarrage ▾

Log in

Print page

## Mission 1 : mise en route

### Introduction

Chacun d'entre vous possède une expérience qui lui est propre en ce qui concerne l'utilisation d'outils informatiques. Certains ont déjà écrit des programmes sur un ordinateur, tandis que d'autres ont à peine effleuré un clavier...

Pour pouvoir commencer à travailler en groupe pour ce cours d'informatique, il est nécessaire de combler quelque peu ce fossé et de vous mettre plus sur un pied d'égalité. Autrement, ce sont ceux qui ont le plus d'expérience qui mèneront les groupes et les plus novices auront moins l'occasion de participer, donc d'apprendre - ce qu'il faut absolument éviter. Nous avons donc organisé une mission de mise en route, que chacun de vous devra accomplir **individuellement**.

#### Deux principes importants

1. Vous devez mener cette mission individuelle de front avec les autres activités de la semaine. A chacun d'entre vous de veiller à combiner travail de groupe et travail individuel, ainsi que travail dans les autres cours. Plus tard, vous devrez être en mesure de combiner encore plus d'activités en y consacrant le temps et les efforts nécessaires.
2. Même si chacun d'entre vous doit avoir effectué la mission individuellement, il est utile que les membres du groupe s'entraident !

Finally, a list with no elements is called an empty list, and is denoted [].

Syllabus  
en ligne :  
exercices

<https://syllabus-interactif.info.ucl.ac.be/>

## Session 2: Somme à compléter

Complétez le fragment de code suivant:

```
# Place dans sum la somme des n premiers entiers pairs

n = some_value
sum = 0
for ___ in _____:
    _____
```

Implémentation

```
1 n = some_value
2 sum = 0
3 for ___ in _____:
4     _____
```

Soumettre >\_

### Informations

Auteur(s)	Tanguy De Bels
Date limite	Pas de date limite
Etat	Pas encore essayé
Note	0.0%
Poids de la note	1.0
Nombre d'essais	0
Limite de soumission	Pas de limite
Étiquettes de catégories	Session 2

### Etiquettes

### Administration

- Voir les soumissions
- Editer l'exercice
- Informations de débogage

### Pour évaluation

- Meilleure soumission
- Pas de soumission

# INGInious

Accès à partir  
du syllabus  
d'exercices

# Exemples résolus avec étapes

## Boucle while

1. Définir et **initialiser** les variables ;
2. Déterminez la **condition** d'arrêt (ou de continuation)
3. Écrire le **corps** de la boucle (mettre à jour la variable de contrôle jusqu'à la condition d'arrêt si nécessaire).

Calculez la somme  $Sum$  des  $n$  premiers entiers positifs

$i = 1$   
 $Sum = 0$

(1) Initialiser

While ( $i \leq n$ )

(2) Condition

$Sum += i$   
 $i += 1$

(3) Corps

<https://syllabus-interactif.info.ucl.ac.be/syllabus/info1-exercices/EXTRA/SLWE>

# A faire dès que possible

---

Si ce n'est déjà fait, **d'urgence**

obtenir son identifiant  
UCLouvain (inscription)

**Ensuite**

S'inscrire au cours sur Moodle  
et INGInious





A faire  
(avant lundi  
prochain)

Lire le syllabus du cours

<https://syllabus-interactif.info.ucl.ac.be/index/info1-theory>

→ chapitres 1 → 5

Faire la mission 1 "mise en route"

<https://syllabus-interactif.info.ucl.ac.be/index/info1-exercises>

→ [Mission 1 - Mise en route](#)

→ [Démarrage](#) : QCM + questions ouvertes

→ [Réalisation](#) d'un programme

# Prochaines échéances

- Avant lundi
  - Avoir lu les chapitres correspondants du syllabus
  - Avoir fait les exercices de démarrage de la mission 1
- Lundi 13h30
  - Séance tutorée intermédiaire Mission 1
- Mercredi 16h00
  - Avoir terminé la phase de réalisation de la mission 1 et soumis le code sur INGIInious
- Jeudi 13h45
  - Séance tutorée finale mission 1
  - Cours de restructuration mission 1
  - Introduction à la matière pour la mission 2