# I. Prise en main du système d'exploitation Windows

## 1. Création de répertoire de travail

a) Visiter votre répertoire où vous avez copié le fichier décompressé « *WinPython-64bit-3.6.3.0Qt5* ».
b) Rendez-vous dans le dossier « *settings* »
c) Créer dans « *settings* » un répertoire « *Programmes MPSI* », avec un sous-répertoire « TP » qui contient un sous-répertoire « TP1 prise en main »
**Remarque :** éviter les accents qui peuvent parfois poser un problème.

## 2. Quelques raccourcis clavier

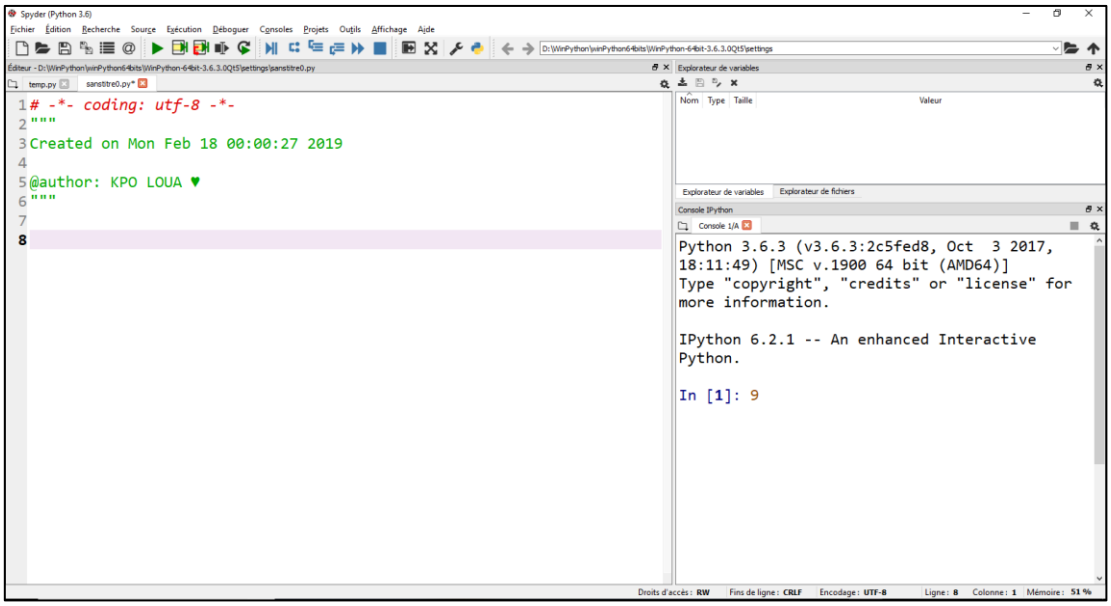
| Tâches | Touches Raccourcis |
|---|---|
| Sauvegarder | Clrt + S |
| Copier (Texte, fichier…) | Clrt + C |
| Coller | Clrt + V |
| Couper (texte, fichier…) | Clrt + X |
| Sélectionner tout | Clrt + A |
| Annuler l'action | Clrt + Z |
| | |

# II. Découverte d'un environnement de développement intégré (IDE)

Dans nos TP nous allons utiliser le logiciel (IDE) Spyder de la distribution complète WinPython.

## 1. Description de la fenêtre

a- Se rendre dans le répertoire « WinPython » et lancer le logiciel « Spyder.exe »
**b- Voir cours ( image N° 5 de la page 11 du cours).**



## 2. Création d'un premier projet

**Exercice :** Codage d'un entier naturel en base k

Représenter en base k un entier naturel exprimé en base 10 (avec k<9). Revoir si nécessaire la méthode vue en cours. L'objectif est d'obtenir l'expression de l'entier n en base k, sous la forme : $a_i$ ….. $a_1$ $a_0$

- Ecrire un algorithme en LDA utilisant la boucle TantQue.

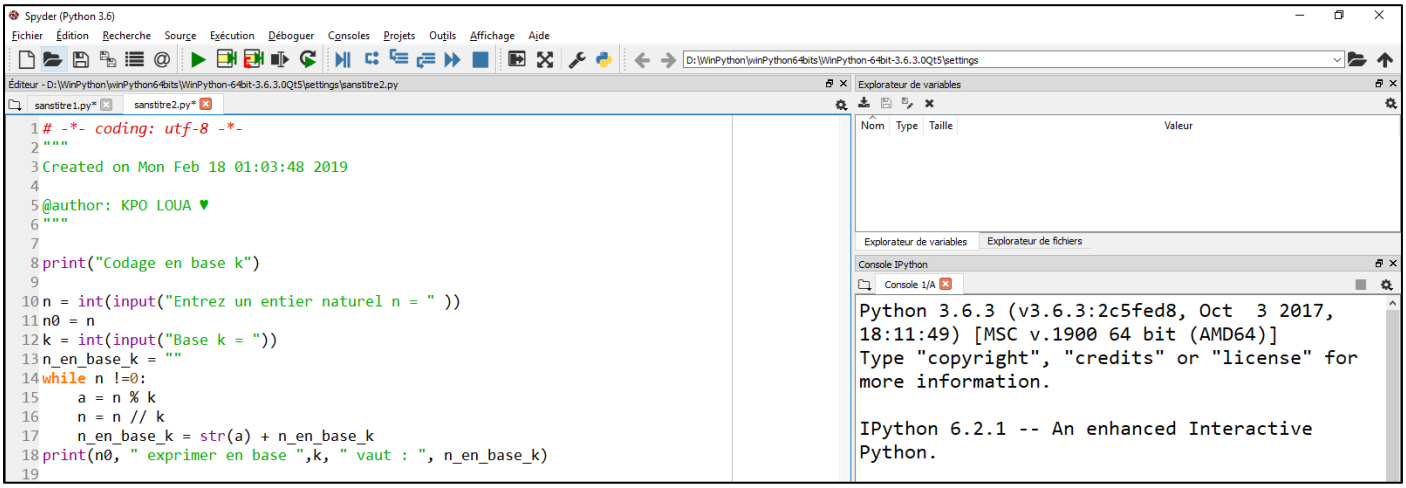
⇒ Traduire l'algorithme en programme informatique écrit dans le langage Python 3.

```
Algorithme Codage_base_k
Var a, n, n0, k : Entier
      n_en_base_k: Chaine
Debut
      Ecrire("Entrez un entier naturel n = " )
      Lire(n)
      n0 = n
      Ecrire("Base k = ")
      Lire(k)
      n_en_base_k = ""
      TantQue n <>0 Faire
            a = n%k
            n = n//k
            n_en_base_k = str(a)+n_en_base_k
      FinTanQue
      Ecrire(n0, " exprimer en base ",k, " vaut : ", n_en_base_k)
Fin
```

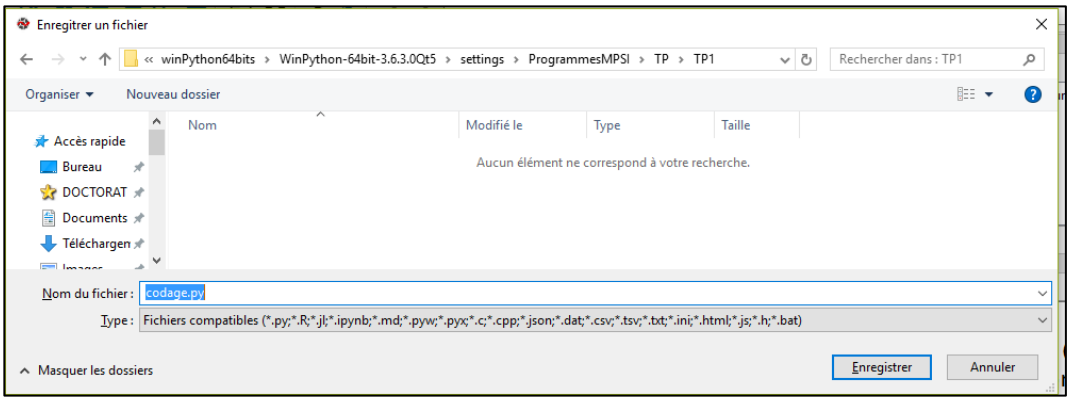
⇒ Saisir le code pour le tester ! Pour cela suivre les instructions suivantes :

- Dans l'IDE Spyder, cliquer sur l'image du menu 
- Placez-vous dans la partie d'édition de texte puis saisissez votre code

```
 8 print("Codage en base k")
 9
10 n = int(input("Entrez un entier naturel n = " ))
11 n0 = n
12 k = int(input("Base k = "))
13 n_en_base_k = ""
14 while n !=0:
15     a = n % k
16     n = n // k
17     n_en_base_k = str(a) + n_en_base_k
18 print(n0, " exprimer en base ",k, " vaut : ", n_en_base_k)
19
20
```

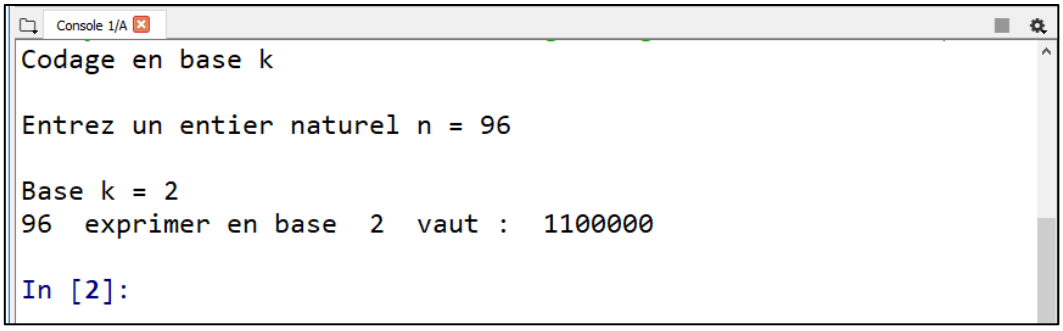



- Sauvegardez votre programme sous le nom de « codage.py » en faisant Clrt + S dans le répertoire « *Programmes MPSI /TP/TP1* »




- Pour le tester appuyez sur la touche « F5 » ou sur l'icône du menu 

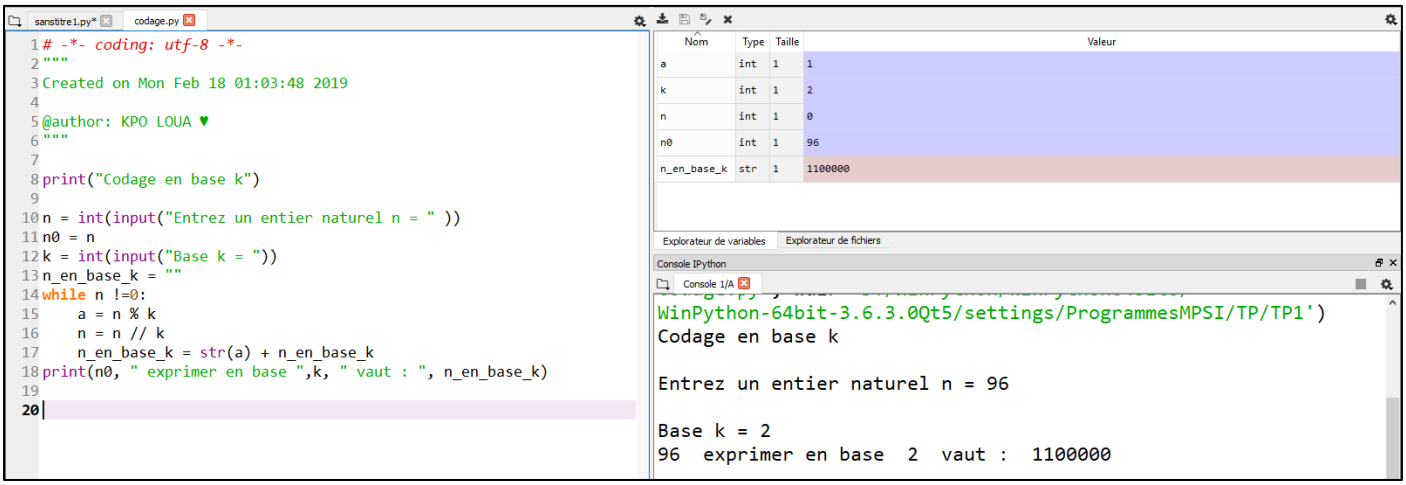
La console se présente comme suit :




L'environnement des variables nous présente toutes les variables qui ont permis d'exécuter ce programme :

| Nom | Type | Taille | Valeur |
|---|---|---|---|
| a | int | 1 | 1 |
| k | int | 1 | 2 |
| n | int | 1 | 0 |
| n0 | int | 1 | 96 |
| n_en_base_k | str | 1 | 1100000 |

### 3. Présentation globale du projet




## III. Travail à faire (TAF)

**TAF :** chaque exercice doit être écrit dans un nouveau fichier d'édition et sauvegarder avec un nom significatif. A la fin de ce TD et TP3 l'étudiant devra avoir crée au moins 12 programmes. Tous les exercices doivent être traités avant le cours prochain !!

### 1. Représentation des nombres

**Exercice 1.** Transformation en binaire d'un nombre exprimé en base 10

Ecrire un programme qui transforme en base 2 un nombre entier positif exprimé en base 10. On pourra utiliser :

- la structure « Tantque » (« while »)
- le quotient de la division euclidienne « // »
- le reste (modulo) de la division euclidienne « % »

**Exemple de résultat attendu :**

⇒ Entrer un entier naturel en base 10 : 12

⇒ « 12 » s'écrit en binaire « 1100 ».

**Exercice 2.** Transformation en base b (b ≤ 9) d'un nombre exprimé en base 10

En modifiant légèrement l'algorithme de l'exercice 1, écrire un programme qui transforme en base b (b ≤ 9) un nombre entier positif exprimé en base 10.

**Exercice 3.** Transformation en base hexadécimale d'un nombre exprimé en base 10

En modifiant légèrement l'algorithme de l'exercice 1, écrire un programme qui transforme en base hexadécimale un nombre entier positif exprimé en base 10. On pourra créer une liste contenant les 16 « symboles » de la représentation hexadécimale.

**Exercice 4.** Transformation en base 10 d'un nombre exprimé en base b (b ≤ 9)

Ecrire un programme qui transforme en base 10 un nombre entier positif exprimé en base b (b ≤ 9). On pourra utiliser deux méthodes :

- la définition ;

- l'algorithme de Hörner, vu en cours.

## 2. if / elif / else

**Exercice 5.** Année bissextile

Ecrire un programme qui permet de savoir si une année (rentrée au clavier) est bissextile.
Définition : Une année est bissextile si elle est :

- soit divisible par 4 mais non divisible par 100 ;
- soit divisible par 400.

**Exercice 6.** Location de voiture

Une agence de location de voiture propose deux formules de location au choix :
1- Location au kilomètre. Le prix de la location se calcule ainsi : 20 €HT / jour

- pour les 100 premiers km : tarif 0.3 €HT du km ;
- pour les kilomètres 101 à 1000 : tarif 0.2 €HT du km ;
- au-delà de 1000 km : tarif 0.1 €HT du km.

2- Forfait journalier, pour lequel le kilométrage est illimité pour un prix fixe de 60 €HT par jour.
Dans les deux cas, il convient d'ajouter une assurance au prix de 5 €HT / jour ainsi que la TVA (20 %).
Ecrire un programme qui, connaissant le nombre de jours de location et le nombre de kilomètres parcourus (données rentrées au clavier), permet de calculer le coût total de la location avec l'une ou l'autre des formules, et de comparer les deux solutions afin de donner la plus avantageuse.

**Exercice 7.** Recherche dans une chaine de caractère

Ecrire un programme qui permet de savoir si une lettre est dans une chaine de caractère. Toutes les deux devront être tapées au clavier et le programme devra afficher le résultat correspondant.
**Exemples de résultats attendus :**

⇒ Rentrer une chaine de caractère : l'année 2019 n'est pas bissextile
⇒ Rentrer un symbole : a
⇒ « a » est dans la chaine de caractère : « l'année 2019 n'est pas bissextile »
⇒ Rentrer une chaine de caractère : l'année 2019 n'est pas bissextile
⇒ Rentrer un symbole : 8
⇒ « 8 » n'est pas dans la chaine de caractère : « l'année 2019 n'est pas bissextile »

**Exercice 8.** Opérations sur deux entiers

Ecrire un programme qui permet de saisir deux entiers et un code opération (+, -, * ou /) et qui affiche le résultat de cette opération sur ces entiers.

## 3. Boucle for

**Exercice 9.** Factorielle n

Ecrire un programme qui permet de calculer « n factorielle (n!)» (n étant un nombre entier positif). On n'utilisera pas le module math !

Exercice 10. Minimum, Maximum et Moyenne d'une suite de n nombres

Ecrire un programme qui demande successivement n nombres à l'utilisateur (n étant donné par l'utilisateur) et qui donne le minimum, le maximum et la moyenne de ces valeurs.

## 4. Boucle while

**Exercice 11.** Filtrage d'un type de nombre

Dans certains programmes, on peut demander à l'utilisateur de rentrer au clavier un type de nombre précis en utilisant la fonction « input() ».
Ecrire un programme qui oblige l'utilisateur à rentrer un entier positif et pas un flottant, des lettres ou tout autre caractère.
**Exemple** de ce que ce programme doit renvoyer comme réponse en fonction de ce qui est tapé au clavier

⇒ Entrer un nombre entier positif : -15.8
⇒ -15.8 n'est pas un entier positif
⇒ Entrer un nombre entier positif : **+@|#
⇒ **+@|# n'est pas un entier positif
⇒ Entrer un nombre entier positif : 16.8
⇒ 16.8 n'est pas un entier positif
⇒ Entrer un nombre entier positif : 1e9
⇒ 1e9 n'est pas un entier positif
⇒ Entrer un nombre entier positif : 26
⇒ le nombre 26 est un nombre entier positif

**Exercice 12.** Recherche du plus petit diviseur d'un entier donné

Ecrire un programme qui recherche le plus petit diviseur autre que 1 d'un entier n donné en paramètre.

**Exercice 13.** Nombre premier

Ecrire un programme qui recherche si n est premier ou non. Indications :

- x est premier si les restes des divisions euclidiennes x / 2, x / 3,..., x / x sont tous non nuls.
- On pourra utiliser la fonction « sqrt » (racine carrée) disponible dans la bibliothèque de fonctions mathématiques de Python. Pour importer cette fonction depuis la bibliothèque « math » de Python, écrire dans la 1ère ligne de votre programme : from math import sqrt

**Exercice 14.** Recherche du PGCD de deux entiers naturels

Etant donnés deux entiers naturels a et b, écrire un programme qui calcule leur plus grand diviseur commun.

- On pourra écrire un premier programme, utilisant les propriétés suivantes du PGCD :
  PGCD (a, b) = PGCD (a - b, b) si a > b
  PGCD (a, b) = PGCD (a, b - a) si b > a
  PGCD (a, a) = a
- On pourra écrire un deuxième programme, utilisant l'algorithme d'Euclide.

**Algorithme d'Euclide :**

Soit a et b deux entiers naturels, si b est non nul, on peut effectuer la division euclidienne de a par b. Il existe un couple unique d'entiers (q, r) tels que : $a = b\,q + r$ et $0 \leq r < b$.
Si r = 0, alors b divise a et PGCD (a, b) = b.
Si r > 0, alors tout diviseur commun à a et b est diviseur de r, et réciproquement, tout diviseur commun à b et r est diviseur de a. On a donc PGCD (a, b) = PGCD (b, r)