I. IMPORTATION ET AFFICHAGE DES IMAGES SOUS PROCESSING

Pour importer une image, il faut utiliser la fonction « loadImage ». L'image doit être chargée dans une variable.

Fonction « loadImage »

Si l'image se trouve dans le dossier du sketch :

```
image = loadImage("nom_fichier.ext")
```

Par exemple,l'image « Bordeaux.jpg » qui se trouve dans le dossier du sketch est importée et chargée dans la variable « img » :

```
img = loadImage("Bordeaux.jpg")
```

Si l'image se trouve sur internet :

```
image = loadImage("url")
```

Par exemple l'image « 14pontpierre_8.jpg » qui se trouve sur le site « www.bordeaux.fr » est importée et chargée dans la variable « img » :

```
img = loadImage("https://lycee-henribrisson.com/wp-
content/uploads/IMG/Logo_Henri_Brisson_Vecto_CMJN_400x319-1.png ")
```

Pour afficher une image importée, il faut utiliser la fonction « image ».

Fonction « image »

```
image("nom_image",x,y)
```

Avec "nom_image" qui représente le nom de la variable dans laquelle a été chargé l'image et x et y les coordonnées de la position dans le fenêtre du coin haut gauche de l'image.

Par exemple, affichage de l'image chargée dans la variable « img » en positionnant le coin haut gauche à la position (0,0) de la fenêtre :

```
image(img,0,0)
```

a) Exercice n°1

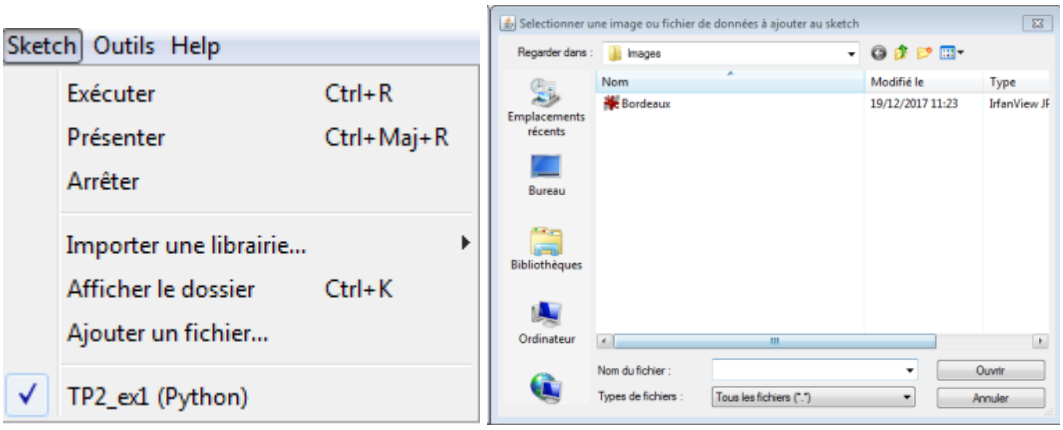
1. Compléter et éditer le sketch suivant permettant d'afficher à partir de la position (0,0) l'image « Bordeaux.jpg ».

```
PImage img;
void setup()
{
        size(400,319) ;
}
void draw()
{
        img = loadImage(……) ;
        image(……,……,……) ;
        noLoop() ;
}
```

2. Enregistrer-le sous « TP1_Ex1 ».
3. Placer l'image «henri-brisson-slider-02.jpg », qui se trouve dans le répertoire « Images » des documents ressource du TP, dans le dossier du sketch.

Pour cela :

- Sélectionner « Ajouter un fichier… » dans le menu « Sketch » de Processing.
- Sélectionner le fichier à placer dans le dossier du « Sketch ».



4. Exécuter le sketch et vérifier son fonctionnement.
5. Modifier le sketch afin que l'image soit centrée dans une fenêtre de taille 600 x 450. Exécuter le sketch et vérifier son fonctionnement.

La fonction « image » peut prendre deux autres paramètres qui permettent de modifier la taille de l'image.

Fonction « image »

image("nom_image",x,y,l,h)

Avec l et h qui représentent la largeur et la hauteur de l'image en pixels.

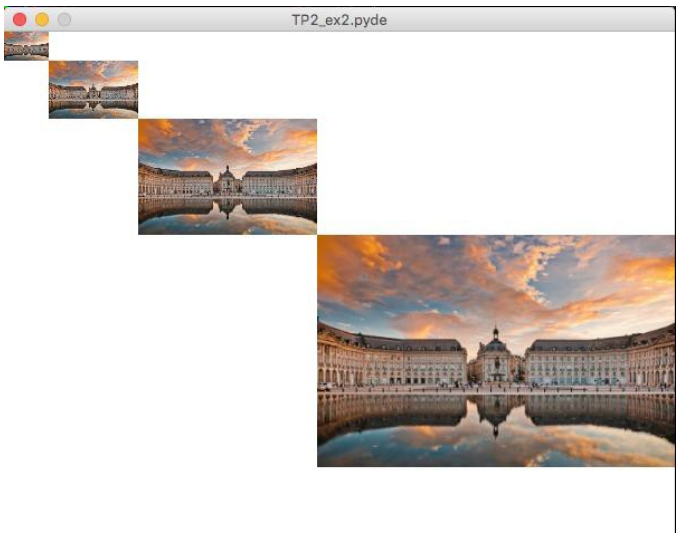
Par exemple affichage de l'image chargée dans la variable « img » en positionnant le coin haut gauche à la position (0,0) de la fenêtre et avec une définition de 200x132 :

image(img,0,0,200,132)

b) Exercice n°2

6. Compléter et éditer le sketch suivant permettant d'obtenir la fenêtre ci-contre. Enregistrer le sous « TP1_Ex2 ».

```
PImage img;
void setup()
{
        size(600,450) ;
        background(255) ;
}
void draw() {
        img = loadImage("henri-brisson-slider-02.jpg ") ;
        image(……,……,……,……,……) ;
        image(……,……,……,……,……) ;
        image(……,……,……,……,……) ;
        image(……,……,……,……,……) ;
}
```



7. Exécuter le sketch et vérifier son fonctionnement.

La fonction « get » permet de sélectionner une partie de l'image.

Fonction « get »

```
nom_image2 = nom_image1.get(x,y,l,h)
```

Avec x et y représentant les coordonnées du point haut gauche de la zone de l'image et l et h la largeur et la hauteur de la zone en pixels.

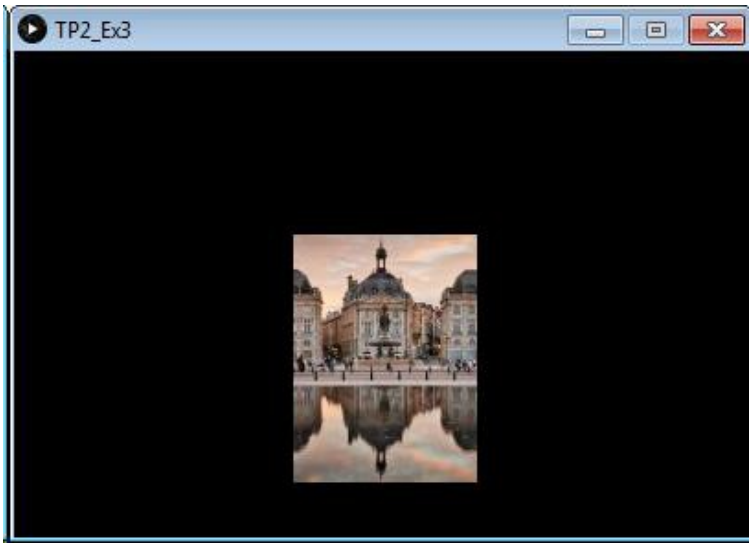
Par exemple :

```
img2 = img.get(img,50,50,50,50)
```

c) Exercice n°3

8. Compléter et éditer le sketch suivant permettant d'obtenir la fenêtre ci-contre. Enregistrer le sous « TP1_Ex3 ».

```
PImage img;
PImage img2;
void setup()
{
        size(600,400) ;
        background(……) ;
}
void draw()
{
        img = loadImage("henri-brisson-slider-02.jpg") ;
        img2 = img.get(……,……,……,……) ;
        image(……,……,……) ;
        noLoop() ;
}
```



9. Exécuter le sketch et vérifier son fonctionnement.

II. COLORISATION DES IMAGES

La fonction « tint » permet de colorer l'image. Elle prend 3 paramètres : la composante rouge, la composante verte et la composante bleue de la couleur désirée. La fonction « noTint » permet d'annuler la colorisation.

Fonctions « tint » et « noTint »

```
tint(rouge,vert,bleu)
```

Par exemple :

```
tint(255,0,0) // colorise l'image en rouge
noTint() // annule la colorisation
```

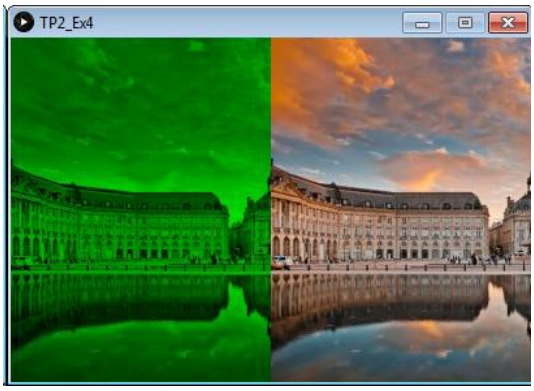
a) Exercice n°4

10. Compléter et éditer le sketch suivant permettant de coloriser l'image en bleu. Enregistrer le sous « TP1_Ex4 ».

```
PImage img;
void setup()
{
        size(400,319) ;
        background(255);
}
void draw()
{
        img = loadImage("henri-brisson-slider-02.jpg");
        tint(……,……,……);
        image(img,0,0) ;
        noLoop();
}
```

b) Exercice n°4 (Suite)

11. Exécuter programme et vérifier son fonctionnement.
12. Modifier le sketch pour obtenir la fenêtre suivante. Exécuter programme et vérifier son fonctionnement.



III. FILTRES

Processing propose quelques filtres de retouche d'image. Ces filtres ne peuvent être appliqués que sur une image déjà affichées. La fonction « filter » permet l'application des filtres. La description des différents filtres dans Processing se trouve à l'adresse : https://processing.org/reference/filter.html.

Fonctions « filter »

```
filter(type, niveau)
```

Avec « type » qui représente le type de filtre et « niveau » l'intensité du filtre. Certains filtres n'ont pas de paramètre « niveau »

L'exemple suivant permet de flouter l'image avec un niveau de 3 :

```
filter(BLUR,3)
```

a) Exercice n°5

13. Compléter et éditer le sketch suivant permettant de créer une image en noir et blanc. Enregistrer le sous « TP1_Ex5 ».

```
void setup()
{
        size(400,319) ;
        background(255);
}
```

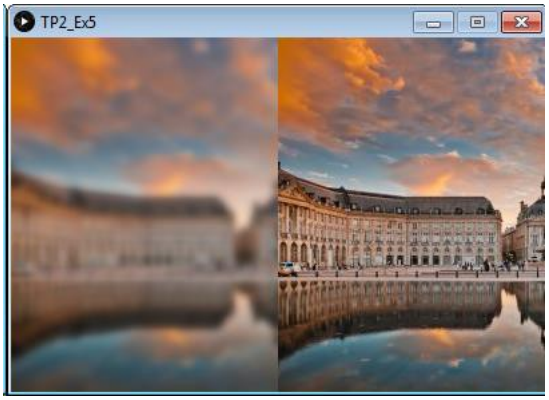
```
void draw()
{
        img = loadImage("henri-brisson-slider-02.jpg ");
        image(img,0,0) ;
        filter(……);
}
```

14. Exécuter programme et vérifier son fonctionnement.

b) Exercice n°5 (Suite)

15. Modifier le sketch pour obtenir la fenêtre suivante. Exécuter programme et vérifier son fonctionnement.

16. Modifier le sketch pour obtenir la fenêtre suivante. Exécuter programme et vérifier son fonctionnement.



IV. ACCES AUX PIXELS

IV. 1. Couleur d'un pixel

Un pixel est le plus petit élément composant une image. Dans Processing, un pixel est un élément de type « color ». La fonction « color » permet d'affecter une couleur RVB à un pixel.

Fonction « color »

```
pixel= color(R, V, B)
```

Avec R, V et B représentant les composantes rouge, verte et bleue de la couleur

Par exemple pour colorier en rouge le pixel1 :

```
pixel1 = color(255, 0, 0)
```

Les fonctions « red », « green » et « blue » permettent de récupérer les composantes rouge, verte et bleue de la couleur d'un pixel.

Fonctions « red », « green » et « blue »

```
composante_rouge = red(pixel)
composante_verte = green(pixel)
```

```
composante_bleue = blue(pixel)
```

IV. 2. Récupérer la couleur des différents pixels constituant une image

Une image est constituée de pixels. Par exemple pour une image de 5x5 :

P0	P1	P2	P3	P4
P5	P6	P7	P8	P9
P10	P11	P12	P13	P14
P15	P16	P17	P18	P19
P20	P21	P22	P23	P24

Fonction « loadPixel »

Il est possible de récupérer la couleur de chacun des pixels constituant l'image en utilisant la fonction « loadPixel ». Elle permet de récupérer la liste de pixels (P0, P1, P2, P3, P4, P5, P6,…., P25). Cette liste prendra le nom « nom_image.pixels ». Pour accéder à un pixel il suffit de placer entre crochet le numéro du pixel.

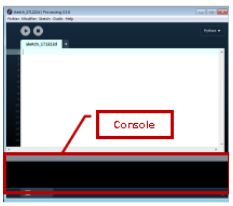
Par exemple pour accéder à la couleur du 1er pixel de l'image «henri-brisson-slider-02.jpg» :

```
img = loadImage("henri-brisson-slider-02.jpg ")
loadPixel()
rouge = red(img.pixels[0])
vert = green(img.pixels[0])
bleu = blue(img.pixels[0])
```

Fonction « print »

La fonction « print » permet d'afficher des données dans la console de Processing. Ces données peuvent être des nombres ou des caractères.

Par exemple pour afficher la composante rouge du 1er pixel de l'image «henri-brisson-slider-02.jpg » :




```
print("Rouge = ",rouge)
```

a) Exercice n°6

Ecrire le sketch suivant qui permet d'afficher l'image « henri-brisson-slider-02.jpg » et de récupérer les composantes rouge, verte et bleue du 10ième pixel de l'image « henri-brisson-slider-02.jpg ». Enregistrer le sous « TP1_Ex6 ».

```
PImage img;
float rouge, vert, bleu;
void setup()
{
        size(400,319) ;
        background(255) ;
}
void draw()
{
        img = loadImage("henri-brisson-slider-02.jpg ") ;
```

```
        image(img,0,0) ;
        loadPixels() ;
        rouge = red(img.pixels[0]) ;
        vert = green(img.pixels[0]) ;
        bleu = blue(img.pixels[0]) ;
        print("Rouge = ",rouge) ;
        print("Vert = ",vert) ;
        print("Bleu = ",bleu) ;
        noLoop();
}
```

17. Exécuter programme et vérifier son fonctionnement.
18. Donner la définition de l'image « henri-brisson-slider-02.jpg» En déduire le numéro du dernier pixel.
19. Modifier le sketch afin d'obtenir les composantes rouge, verte et bleue du dernier pixel de l'image « henri-brisson-slider-02.jpg ». Exécuter programme et vérifier son fonctionnement.

IV. 3. Modifier la couleur de pixels constituant une image

Pour modifier la couleur d'un pixel il faut utiliser la fonction « color » suivi de la fonction « updatePixels » après la modification et avant l'affichage

Fonction « updatePixels »

```
pixel= color(R, V, B)
updatePixels()
```

Par exemple pour colorier en rouge le pixel1 :

```
pixel1 = color(255, 0, 0)
updatePixels()
```

a) Exercice n°7

Editer le sketch suivant qui permet d'afficher l'image « henri-brisson-slider-02.jpg » en noir et blanc. Enregistrer le sous « TP1_Ex7 ».

```
PImage img;
int N, i;
void setup()
{
        size(400,319) ;
        background(255) ;
}
void draw()
{
        img = loadImage("henri-brisson-slider-02.jpg") ;
        loadPixels() ;
        N = img.width * img.height;
        print(N);
        for (i=0 ;i<N ;i++)
```

```
        {
                if (color(img.pixels[i]) < color(127,127,127))
                {
                        img.pixels[i] = color(0,0,0) ;
                }
                else
                {
                        img.pixels[i] = color(255,255,255) ;
                }
                updatePixels() ;
        }
        image(img,0,0) ;
        noLoop() ;
}
```

20. Modifier dans la ligne « if (color(img.pixels[i])<color(127,127,127)) : » la valeur 127 par la valeur 200. Exécuter le programme.
21. Modifier dans la valeur 127 par 80. Exécuter le programme.

b) Exercice n°7 (Suite)

22. Modifier le sketch afin d'obtenir l'image suivante. Exécuter programme et vérifier son fonctionnement.