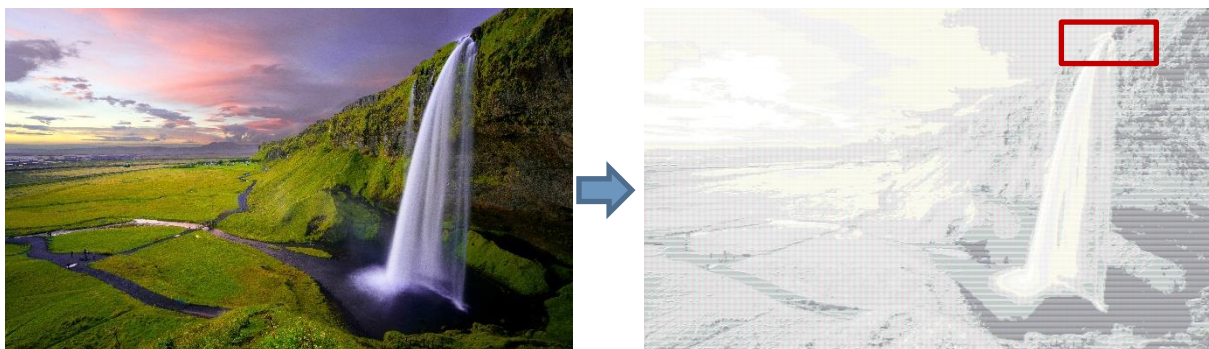


SAÉ S1.01 – Implémentation d'un besoin client

Présentation du contexte

L'Ascii Art est l'art de représenter des images uniquement avec des lettres ou des caractères spéciaux. Votre client souhaite disposer d'un logiciel lui permettant de retranscrire des images en Ascii Art, tel que sur l'exemple ci-dessous :



Un zoom sur le haut de la cascade montre que l'image est en réalité formée de caractères :

[illegible]

Votre objectif est de développer un logiciel capable de réaliser ce traitement.

👉 L'organisation du sujet de cette SAÉ est itérative, de versions en versions du logiciel. Plus vous atteindrez une version élevée, plus votre travail sera récompensé.

 IL EST FORTEMENT CONSEILLE DE LIRE L'ENSEMBLE DE CE DOCUMENT AVANT DE COMMENCER A CODER !

Version 1 La base

Un utilitaire *img2pgm* vous est fourni. Il permet de transformer des fichiers images du format PNG ou JPEG au format PGM. Le format PGM est un format de fichier que vous pourrez lire plus facilement dans votre programme. Avant d'étudier le format PGM, commencez par prendre en main l'utilitaire *img2pgm*.

1.1 Mode d'emploi de *img2pgm*

L'utilitaire *img2pgm* s'utilise en ligne de commande. Copiez l'utilitaire dans un répertoire de votre disque dur (par exemple *C:\img2pgm*) ainsi que les fichiers images que vous souhaitez convertir en PGM (par exemple *image1.png* et *image2.jpg*). Ouvrez une invite de commandes et naviguez vers ce répertoire puis entrez la commande **img2pgm image1.png**. Cela crée dans le même répertoire le fichier *image1.pgm*, qui est la version au format PGM de l'image originale.

D'autres options de conversion sont disponibles. Entrez la commande **img2pgm --help** pour avoir l'aide en ligne de cet utilitaire. Voici son résultat :

```
Usage :  
img2pgm [-i] fichier_image [-o fichier_image]  
  
Options disponibles :  
-h [ --help ]           message d'aide  
-i [ --input ] arg      fichier image à convertir (format png ou jpg)  
-o [ --output ] arg     fichier image de sortie (format PGM)  
  
En l'absence d'option --output, le fichier de sortie porte le même nom  
que le fichier d'entrée mais avec l'extension .pgm.
```

Sachant que l'argument *'-i'* est par défaut, vous pouvez également de façon plus pratique faire glisser un fichier image sur l'icône de l'utilitaire *img2pgm* dans l'explorateur Windows. Cela créera dans le répertoire du fichier image sa version convertie en PGM.

Ces fichiers PGM peuvent être visualisés par de nombreux logiciels de traitement d'image, comme [Gimp](#).

1.2 Le format PGM

Si vous voulez tout savoir sur le format PGM, [Wikipédia](#) est votre ami. En ce qui concerne l'utilitaire *img2pgm*, le format du fichier produit est le format PGM binaire sur 256 niveaux de gris. Les fichiers de sortie de *img2pgm* sont constitués de la façon suivante :

- Un entête de fichier constituée elle-même des informations suivantes :
 - 2 caractères magiques « **P5** »
 - 1 saut de ligne
 - Une ligne de commentaire commençant par un **#**. Cette ligne est à ignorer.
 - 1 saut de ligne
 - La largeur de l'image stockée en texte, un espace, la hauteur de l'image stockée en texte
 - 1 saut de ligne
 - La valeur 255 stockée en texte

- 1 saut de ligne
- Les données binaires de l'image :
 - L'image est décrite ligne par ligne en partant du haut
 - Chaque ligne est codée de gauche à droite.
 - Chaque pixel est codé sur un octet.

D'un point de vue programmation, la lecture de l'entête se fait de façon habituelle à l'aide d'une variable de type `std::ifstream`. Cependant, la lecture des données binaires qui suivent doivent être lues en utilisant la fonction `read()` et nécessite que le fichier soit ouvert en *mode binaire*. Ainsi, le code suivant vous montre un exemple d'ouverture d'un fichier en mode binaire, puis de la lecture de 10 octets qui seront stockés dans un tableau :

```
//Ouverture du fichier "ImageTest1.pgm" en mode binaire :
std::ifstream fichier("ImageTest1.pgm", std::ios_base::binary);

//Ici vous devez décoder l'entête du fichier
//...

//Création d'une mémoire de 10 octets :
std::vector<char> donnees(10);



//Lecture de 10 octets depuis le fichier et stockage dans le tableau donnees :
fichier.read(donnees.data(), 10);
```

1.3 La transcription des données PGM en Ascii Art

Chaque pixel de l'image est stocké dans le fichier PGM à l'aide d'un octet dont la valeur varie de 0 (noir) à 255 (blanc). En Ascii Art, les nuances de gris sont représentées par des caractères plus ou moins imposants. Par exemple, un « W » semble plus sombre qu'un « . » car il « remplit » plus sa cellule. Ainsi, votre client vous propose de retranscrire les nuances de gris à l'aide de la palette de caractères définie dans le tableau suivant :

W	w	l	i	:	,	.	espace

1.4 À vous de jouer

-  En vous appuyant sur les informations précédentes, implémentez un programme qui lira un fichier PGM et le retranscrira en Ascii Art dans la console.
-  Il serait souhaitable que le nom du fichier PGM soit renseigné par l'utilisateur.

Version 2 Enregistrement de l'Ascii Art

Félicitations, votre client est enchanté par votre première version. Il est prêt à vous financer une version 2 dans laquelle il serait possible d'enregistrer l'Ascii Art dans un fichier texte. Cela serait en effet plus pratique pour des images de plus grande taille que la limite de 150 caractères de large environ de la console.

- 📁 Implémentez cette nouvelle fonctionnalité
 - ➡ Il serait souhaitable que le nom du fichier texte de sortie soit renseigné par l'utilisateur.
 - ➡ N'oubliez pas de tester votre programme en affichant le fichier texte dans un éditeur capable d'afficher des polices de très petite taille (par exemple *notepad++* en zoomant au minimum).

Version 3 Choix de la palette

Une nouvelle fois, votre client apprécie votre travail et surtout la vitesse à laquelle vous avez apporté cette modification. Cette fois, il souhaite que vous apportiez la possibilité de changer de palette de caractères par le choix d'un fichier de palette. Les fichiers de palette sont des fichiers texte dont chaque ligne est une chaîne de caractères représentant un niveau de gris. La première ligne code le niveau de gris le plus sombre, la dernière le plus clair. Quelques exemples de palettes vous ont été gracieusement fournies par votre client afin de vous aider dans vos tests.

- 📁 Implémentez cette nouvelle fonctionnalité

Version 4 Arguments de la ligne de commande

Plus rien ne vous arrête ! Votre client souhaiterait que la configuration de votre programme ne se fasse plus par la saisie des noms de fichiers dans la console en cours d'exécution du programme, mais par un paramétrage à l'aide d'arguments de votre programme. Il vous propose alors que votre programme puisse être exécuté à l'invite de commande avec les options suivantes (en supposant que votre programme s'appelle *pgm2txt*) :

```
Usage :
pgm2txt [options]

Options :
--input fichier      Spécifie le fichier image à convertir
                     Si ce paramètre n'est pas spécifié, le fichier est demandé via la
                     console.

--output fichier     Spécifie le nom du fichier texte qui contiendra l'Ascii Art.
                     Si ce paramètre n'est pas spécifié, l'Ascii Art est sortie dans la
                     console.

--palette fichier    Spécifie un fichier texte contenant la palette de couleur Ascii.
                     Chaque ligne du fichier contient un caractère en UTF-8, du plus
                     sombre au plus clair.
                     Si ce paramètre n'est pas spécifié, la palette par défaut est
                     "W", "w", "l", "i", ":", ",", ".", " "

--help              Affiche cette aide.
```

4.1 Comment passer des arguments à un programme ?

Là, vous êtes pris de panique ! Mais le C++, comme souvent, a déjà pensé à vous. La fonction `main()` de votre programme peut disposer de deux paramètres : le premier est un entier qui est le nombre de paramètres de la ligne de commande et le second est un tableau de chaîne de caractères contenant les paramètres. Ainsi, le code suivant permet d'afficher tous les arguments d'un programme :

```
int main(int argc, char* argv[])
{
    for (int i = 0; i <= argc; ++i)
        std::cout << i << " : " << argv[i] << '\n';

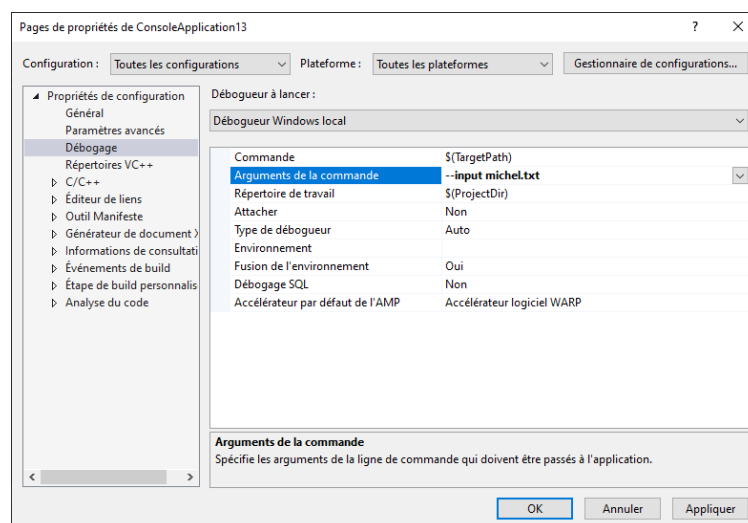
    return 0;
}
```

S'il se nomme **prog.exe** et est exécuté avec les arguments « **--input michel.txt** », alors sa sortie sera :

```
C:\Programme>prog.exe --input michel.txt
0 : prog.exe
1 : --input
2 : michel.txt
3 :
```

4.2 Débugger un programme qui prend des arguments dans Visual Studio

Exécuter votre programme depuis la ligne de commande n'est pas forcément pratique quand vous êtes en train de le programmer dans Visual Studio. Il est alors possible de paramétrer votre projet pour que le programme s'exécute depuis Visual Studio directement avec les arguments souhaités. Pour cela, dans les propriétés du projet, vous pouvez ajouter tous les arguments de commandes nécessaires dans le champ *Débogage / Arguments de la commande*, comme représenté dans la figure ci-dessous :



4.3 Codez !

 Implémentez ce besoin exprimé.

Version 5 Réduction de la taille des images

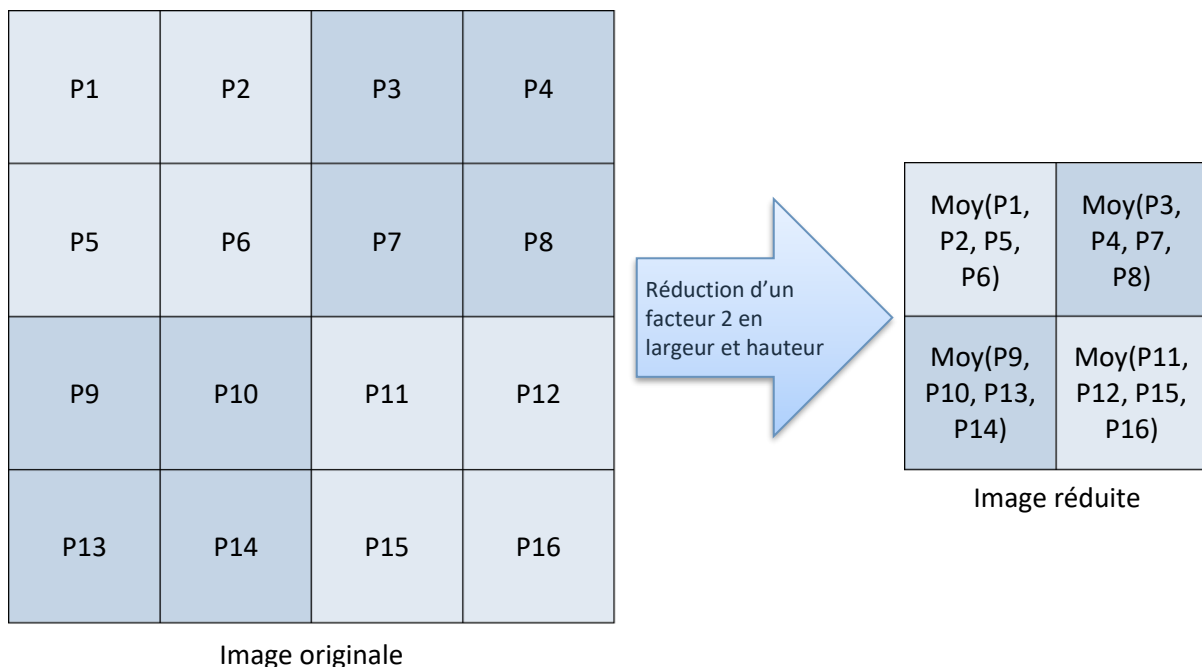
Vous avez réussi à fournir la version 4, bravo ! Cette fois-ci le client rencontre un petit problème de taille. Dans l'état actuel, votre programme retranscrit directement chaque pixel de l'image initiale en caractères selon la palette. Cela fonctionne très bien, mais dans le cas de grandes images, cela fait des très grand fichiers texte. Il souhaiterait que vous apportiez deux arguments supplémentaires à votre programme :

```
--width nombre    Spécifie la largeur max de l'Ascii Art.
                  Si ce paramètre n'est pas spécifié, aucune limite n'est fixée.
                  Voir Remarques.


--height nombre   Spécifie la hauteur max de l'Ascii Art.
                  Si ce paramètre n'est pas spécifié, aucune limite n'est fixée.
                  Voir Remarques.

Remarques :
Quelles que soient les valeurs des options --width et --height, la taille de l'Ascii Art
est bornée par la taille de l'image en entrée. La taille de l'ascii art conserve
toujours le même ratio que l'image d'entrée, les valeurs des options --width et --height
ne sont que des maximums.
```




Votre logiciel doit donc maintenant être capable de réduire la taille des images. Cela s'appelle du « rééchantillonnage ». Dans le cas d'une réduction, cela consiste à calculer chaque pixel de l'image réduite comme étant une moyenne des pixels correspondants dans l'image originale. Cela peut se représenter schématiquement :



La stratégie de l'algorithme à mettre en œuvre est de parcourir l'ensemble des pixels de l'image réduite et pour chacun, de déterminer en fonction du facteur de réduction, l'ensemble des pixels correspondants dans l'image originale et d'en calculer la moyenne.

 Implémentez cette fonctionnalité.

Remise de votre travail

-  Remettez dans l'espace du cours en ligne un fichier compressé contenant :
 -  Votre code source en ayant pris soin de supprimer l'inutile
 -  Un fichier PDF d'un rapport succinct présentant la version que vous avez atteinte ainsi que les tests que vous avez effectués (capture d'écran à l'appui)