

# OCR Word Search Solver

## Cahier des charges

### Table des matières

I. Le cadre.....	3
II. Les dates clés.....	3
III. Le sujet.....	3
1. Description générale.....	3
2. Description détaillée.....	4
2.1. Principales étapes.....	4
2.2. Découpage de l'image.....	4
2.3. Reconnaissance de caractères (réseau de neurones).....	4
2.4. Prétraitement.....	5
2.5. Résolution d'une grille de mots cachés.....	5
2.6. Format d'affichage de la grille résolue.....	5
IV. Les contraintes du projet.....	6
1. Les contraintes techniques.....	6
2. Gestion de versions et procédure de rendu.....	6
3. Qualité des images à traiter.....	6
3.1. Niveau 1.....	6
3.2. Niveau 2.....	7
3.3. Niveau 3.....	7
3.4. Niveau 4.....	7
V. Les livrables.....	7
1. Le plan de soutenance.....	7
2. Le rapport en version papier.....	7
3. Le contenu du dépôt GIT.....	8
3.1. Le rapport en version numérique.....	8
3.2. Le code.....	8
3.3. Les fichiers README et AUTHORS.....	8
3.4. Les images.....	8
VI. Les soutenances.....	9
1. Structure générale.....	9
2. Découpage du projet.....	9
3. Notation.....	10
VII. Quelques conseils.....	10
VIII. Annexes.....	11
1. Formats utilisés par le programme solver.....	11
2. Exemples d'images.....	12
2.1. Niveau 1.....	12
2.2. Niveau 2.....	13

2.3. Niveau 3.....	14
2.4. Niveau 4.....	15
2.5. Liens de téléchargement.....	16

## I. Le cadre

Le projet est à réaliser en groupe de quatre personnes (et uniquement quatre). Il s'étend du mois de septembre au mois de décembre.

Ce document présente les objectifs du projet, les différentes parties attendues, le planning de réalisation et ses différentes contraintes.

## II. Les dates clés

- **Vendredi 20 septembre 2024** : date limite de constitution des groupes (le formulaire sera ouvert du 16 au 20 septembre).
- **Semaine du 4 novembre 2024** : première soutenance (éventuellement en fin de journée).
  - Les dépôts GIT seront clonés le dimanche 3 novembre à 20 h.
- **Semaine du 9 décembre 2024** : soutenance finale (éventuellement en fin de journée).
  - Les dépôts GIT seront clonés le dimanche 8 décembre à 20 h.

## III. Le sujet

### 1. Description générale

L'objectif de ce projet est de réaliser un logiciel de type OCR (*Optical Character Recognition*) qui résout une grille de mots cachés.

Pour plus de détails sur les grilles de mots cachés, vous pouvez consulter la page suivante :

[https://en.wikipedia.org/wiki/Word\\_search](https://en.wikipedia.org/wiki/Word_search)

Votre application prendra donc en entrée une image représentant une grille de mots cachés et affichera en sortie la grille résolue.

Dans sa version définitive, votre application devra proposer une interface graphique permettant de charger une image dans un format standard, de la visualiser, de corriger certains de ses défauts, et enfin d'afficher la grille complètement remplie et résolue. La grille résolue devra également pouvoir être sauvegardée.

Votre application devra aussi posséder un aspect apprentissage, qui pourra être séparé de la partie principale, et qui permettra d'entraîner votre réseau de neurones, puis de sauvegarder et de recharger le résultat de cet apprentissage.

## 2. Description détaillée

### 2.1. Principales étapes

Le traitement effectué par votre application sera **approximativement** le suivant :

- Chargement d'une image ;
- Suppression des couleurs (niveau de gris, puis noir et blanc) ;
- Prétraitement ;
- Détection de la position grille ;
- Détection de la position de la liste de mots ;
- Détection des cases de la grille ;
- Récupération sous la forme d'image :
  - Des lettres présentes dans les cases ;
  - Des mots de la liste ;
  - Des lettres qui constituent les mots de la liste.
- Reconnaissance de caractères à partir des images :
  - Des lettres présentes dans les cases ;
  - Des lettres qui constituent les mots de la liste.
- Reconstruction de la grille sous la forme d'un tableau de caractères ;
- Reconstruction de la liste de mots sous la forme d'un tableau de chaînes de caractères ;
- Résolution de la grille ;
- Affichage de la grille résolue ;
- Sauvegarde de la grille résolue.

On notera qu'aucune information de couleur n'est nécessaire pour la reconnaissance de caractères et qu'il est même fortement conseillé de travailler en noir et blanc.

### 2.2. Découpage de l'image

Le découpage de l'image est nécessaire pour envoyer à la reconnaissance les morceaux d'images correspondants à des lettres.

Il faudra donc identifier la position des lettres (dans les cases de la grille et dans les mots de la liste) afin d'extraire les lettres sous la forme d'image.

### 2.3. Reconnaissance de caractères (réseau de neurones)

Cette partie nécessite une phase d'apprentissage pendant laquelle votre réseau de neurones va apprendre à reconnaître les différents caractères.

Un réseau de neurones est un outil permettant d'apprendre une fonction (possiblement non linéaire) à l'aide d'exemples. Dans la phase d'apprentissage du réseau, vous allez fournir des entrées déjà identifiées (ici des blocs d'images) auxquelles votre réseau répondra. La marge d'erreur de ces réponses sera alors utilisée pour corriger les poids internes du réseau. Avec un certain nombre d'exemples, votre réseau finira,

dans la majorité des cas, par fournir les bonnes réponses. La sortie usuelle pour ce genre de problème est une probabilité par symbole du jeu de caractères. Le symbole avec la plus forte probabilité sera retenu comme le symbole reconnu.

Pour implémenter votre réseau de neurones, voici quelques termes à rechercher : **apprentissage supervisé, perceptron multicouche, rétropropagation, descente de gradient, AdaBoost.**

Le site <http://neuralnetworksanddeeplearning.com> (en anglais) fournit beaucoup d'informations théoriques et pratiques qui pourraient vous être utiles.

Il existe de nombreuses formes de réseaux de neurones, mais pour votre projet vous n'avez probablement pas besoin de plus d'une couche cachée. Par contre, il peut être judicieux de remplacer la fonction d'activation de la dernière couche par un *softmax* plutôt que par la fonction sigmoïde habituelle. Vous trouverez toutes ces informations dans le lien fourni plus haut.

## 2.4. Prétraitement

L'objectif de cette partie est d'améliorer la qualité des images traitées. Cela permettra à votre application d'accepter en entrée des documents directement issus d'un scanner ou d'une photo. Voici les prétraitements que vous devez réaliser :

- Redressement manuel de l'image (rotation de l'image à partir d'un angle saisi par l'utilisateur) ;
- Redressement automatique de l'image (détection d'angle de rotation puis rotation de l'image) ;
- Élimination de bruits parasites (grain de l'image, tâches, etc.) ;
- Renforcement des contrastes.

## 2.5. Résolution d'une grille de mots cachés

Vous devrez implémenter un algorithme de résolution de mots cachés en langage C. Pour tester cet algorithme, vous devrez concevoir un petit programme appelé **solver**. Ce sera une application qui s'exécutera en ligne de commande uniquement. Il prendra en entrée un nom de fichier et un mot. Le fichier contiendra une grille dans un format donné en annexe. Le mot (second argument) devra être trouvé dans la grille. L'application affichera sur la sortie standard la position dans la grille de la première lettre et de la dernière lettre du mot. Le format d'affichage est imposé et détaillé dans l'annexe.

## 2.6. Format d'affichage de la grille résolue

Le résultat s'affichera sous la forme d'une image. Cette image pourra être sauvegardée dans un format d'image standard.

Au minimum, l'image contiendra la grille avec les mots cachés encerclés.

**La qualité de l'affichage sera prise en compte dans l'évaluation.**

## IV. Les contraintes du projet

### 1. Les contraintes techniques

Vous devrez respecter les points suivants :

- Votre projet sera écrit en langage C ;
- Il devra compiler et s'exécuter dans l'environnement de travail qui vous est fourni par l'école ;
- Lors des soutenances, les démonstrations se feront sur les machines de l'école ;
- Tous les logiciels installés sur les machines de l'école sont autorisés à être utilisés pour le projet ;
- Votre code devra compiler sans erreur avec au moins les options suivantes : -Wall -Wextra ;
- Les lignes de code ne devront pas dépasser 80 colonnes et ne devront pas contenir d'espaces inutiles en fin de ligne ;
- Tous les identifiants utilisés dans votre code ainsi que les commentaires devront être en anglais ;
- Votre dépôt devra contenir **uniquement** les fichiers précisés dans la partie V.3 *Le contenu du dépôt GIT* ;
- **Votre dépôt ne devra pas contenir :**
  - De fichiers exécutables ;
  - De code source extérieur (bibliothèques, autres projets, etc.) ;
  - Tout autre type de fichier non nécessaire au projet.

### 2. Gestion de versions et procédure de rendu

Pour le projet, vous disposerez d'un dépôt *git* utilisable par tous les membres d'un groupe. Il est fortement conseillé d'utiliser les fonctionnalités de gestion de version de *git*.

Pour chaque soutenance, un rendu de code vous sera demandé avant la soutenance. Ce rendu de code s'effectuera à l'aide de *git* : les examinateurs cloneront votre dépôt sur la branche principale (*master*) à la date et à l'heure prévues pour le rendu. **Seuls les fichiers présents sur votre dépôt au moment du clonage seront disponibles pendant la soutenance.**

### 3. Qualité des images à traiter

Les images à traiter pourront être variées et de différentes qualités. Nous classerons ces images selon différents critères qui pourront s'organiser en quatre niveaux de difficulté.

#### 3.1. Niveau 1

- L'image ne contient que la grille et la liste de mots ;
- L'image ne comporte aucun défaut ;
- Toutes les lettres sont en majuscules ;
- La grille ou les cases de la grille peuvent être délimitées ou non par une bordure ;
- Les cases de la grille sont de tailles identiques ;
- La grille et la liste peuvent se trouver n'importe où sur l'image ;

- La taille de la grille est aléatoire et contient au minimum cinq lignes et cinq colonnes ;
- L'image peut être en couleur ou en noir et blanc ;
- La liste de mots est sur une seule colonne et contient un seul mot par ligne.

### **3.2. Niveau 2**

En plus du niveau 1, l'image peut comporter certains défauts comme :

- Un mauvais contraste ;
- Des ombres ;
- Du bruit parasite ;
- Des caractères ou des lignes flous ;
- Une rotation d'un angle aléatoire.

### **3.3. Niveau 3**

En plus des défauts du niveau 2 :

- L'image peut contenir autre chose que la grille et la liste de mots ;
- La liste de mots peut être sur plusieurs colonnes.

### **3.4. Niveau 4**

En plus du niveau 3 :

- Les lettres des mots de la liste peuvent être en minuscules ;
- Les cases de la grille peuvent être de tailles différentes.

Vous trouverez quelques images illustrant ces quatre niveaux de difficulté dans l'annexe.

## **V. Les livrables**

### **1. Le plan de soutenance**

Un plan de soutenance sera à fournir au début de chaque soutenance. Il sera au format papier et devra tenir sur le recto d'une feuille A4.

### **2. Le rapport en version papier**

Tout comme le plan de soutenance, un rapport présentant votre projet devra être imprimé et rendu le jour de la soutenance.

Il devra, entre autres, présenter :

- Votre groupe ;
- La répartition des charges et l'état d'avancement du projet ;
- Les aspects techniques.

### 3. Le contenu du dépôt GIT

#### 3.1. Le rapport en version numérique

Une version numérique de votre rapport devra également être présente dans votre dépôt GIT. Elle devra être au format PDF et se situer à la racine de votre dépôt. Le fichier devra porter le nom suivant :

- **rapport\_1.pdf** pour la première soutenance (15 pages minimum) ;
- **rapport\_2.pdf** pour la soutenance finale (40 pages minimum).

**Attention, la récupération des rapports sur le dépôt GIT sera automatisée. Il est donc important de respecter les noms de fichiers ci-dessus, faute de quoi votre rapport ne sera pas évalué.**

#### 3.2. Le code

Votre dépôt GIT devra contenir l'intégralité du code source de votre projet ainsi qu'un fichier **Makefile**. Ce dernier devra fournir au moins les règles *all* et *clean*.

#### 3.3. Les fichiers README et AUTHORS

Votre dépôt GIT devra également contenir :

- Un fichier **README** décrivant de manière concise comment utiliser votre application. Ce fichier devra être au format texte et pouvoir être lu correctement dans un terminal avec une commande comme *cat* ou *less*. Il pourra éventuellement être au format *markdown* ;
- Un fichier **AUTHORS** contenant les logins des membres de votre groupe. Il devra contenir une ligne par membre du groupe sous la forme suivante : login (NOM Prénom).

#### 3.4. Les images

Votre dépôt GIT devra aussi contenir quelques images qui pourront servir aux démonstrations de vos différentes parties. Attention, ne surchargez pas votre dépôt avec un grand nombre d'images à très haute résolution. **Trois ou quatre images à une résolution correcte devraient suffire.**

**Il vous faudra également un jeu d'images pour l'apprentissage de votre réseau de neurones.**



## VI. Les soutenances

### 1. Structure générale

Pour chaque soutenance, vous devrez prévoir une introduction, une conclusion et une démonstration de toutes les fonctionnalités attendues. Étant donné que le sujet est commun à tous les groupes, il n'est pas utile de le présenter ni de paraphraser le présent document.

La première soutenance est une soutenance de suivi. L'objectif est de montrer votre état d'avancement. Vous devrez également expliquer le fonctionnement de votre réseau de neurones. Attention, là encore, il ne vous est pas demandé de faire un exposé sur les réseaux de neurones, mais bien de décrire le fonctionnement de celui que vous avez implémenté.

Pour la soutenance finale, vous devrez faire la démonstration d'un programme fini. Il est important que celui-ci soit homogène et permette d'effectuer la chaîne de traitements attendus. Il est plus important d'avoir un programme complet (et pas une collection de bouts de code) même si celui-ci ne fonctionne que sur un jeu réduit d'exemples.

### 2. Découpage du projet

Votre projet se découpera en plusieurs éléments. Un élément est considéré comme étant terminé si votre programme est capable d'effectuer la tâche attendue sur quelques exemples non triviaux. Bien évidemment, l'évaluation complète de l'élément dépendra de sa qualité et de sa robustesse face à des tests avancés.

#### Éléments devant être présentés et obligatoirement terminés pour la première soutenance :

- Chargement d'une image et suppression des couleurs ;
- Rotation manuelle de l'image ;
- Détection de la position :
  - De la grille ;
  - De la liste de mots ;
  - Des lettres dans la grille ;
  - Des mots de la liste ;
  - Des lettres dans les mots de la liste.
- Découpage de l'image (sauvegarde de chaque lettre sous la forme d'une image) ;
- Implémentation de l'algorithme de résolution d'une grille de mots cachés. Vous devrez implémenter cet algorithme dans le programme en ligne de commande **solver** (cf. III.2.5 Résolution d'une grille de mots cachés) ;
- Une preuve de concept de votre réseau de neurones. Pour cette preuve, vous réaliserez un mini réseau capable d'apprendre la fonction logique  $\overline{A}.\overline{B} + A.B$  (A et B étant des variables booléennes n'acceptant que les valeurs 0 et 1).

### Éléments devant être présentés et obligatoirement terminés pour la soutenance finale :

- Le prétraitement complet (avec la rotation automatique) ;
- Le réseau de neurones complet et fonctionnel
  - Apprentissage ;
  - Reconnaissance des lettres de la grille et de la liste de mots.
- La reconstruction de la grille ;
- La reconstruction de la liste de mots ;
- La résolution de la grille ;
- L’affichage de la grille ;
- La sauvegarde du résultat ;
- Une interface graphique permettant d’utiliser tous ces éléments.

### Éléments supplémentaires (évalués uniquement si la partie obligatoire est faite) :

- Générateur d’une grille de mots cachés à partir d’une liste de mots ;
- Site internet présentant votre projet ;
- Autres.

## 3. Notation

Voir les grilles critériées sur Moodle.

## VII. Quelques conseils

Ce projet est conçu pour être réalisé en équipe de quatre personnes sur une période totale de trois mois. Certains éléments, comme le réseau de neurones, nécessitent des phases d’ajustement et parfois des phases de calcul pouvant prendre du temps. **Il est donc important de commencer à travailler dès la constitution de votre groupe, sous peine de ne pas disposer d’assez de temps pour finaliser votre projet.**

Il peut être également judicieux de concevoir des tests indépendants pour chaque partie. Par exemple, le réseau de neurones devra recevoir en entrée des images de caractères déjà découpées et redimensionnées. Afin de tester et d’entraîner votre réseau de neurones, il serait pertinent de disposer d’un jeu de tests où les images sont déjà des caractères seuls au bon format. Ainsi, même si votre découpage de l’image n’est pas prêt, vous pourrez faire avancer cette partie.

Lors de la première soutenance, nous vous demanderons de faire la démonstration des parties attendues. Pensez à préparer votre projet pour faire la démonstration de ces parties de manière simple et efficace (**visualisation intermédiaire**, exécutions indépendantes, etc.).

Il y a quelques écueils en C qui pourraient vous faire perdre beaucoup de temps. Vous aurez besoin de porter une attention particulière à de nombreux détails techniques. Prenez de bonnes habitudes pour éviter de perdre du temps. Notamment, pour tous vos problèmes de mémoire, utilisez un outil comme *valgrind* ou l’option `-fsanitize=address` de *gcc*.

## VIII. Annexes

### 1. Formats utilisés par le programme **solver**

Le programme **solver** prendra deux arguments en entrée :

- Le nom d'un fichier texte. Ce fichier contiendra une grille de caractères en majuscule. Cette grille sera constituée au minimum de cinq lignes et de cinq colonnes.
- Le mot à chercher dans la grille (en majuscules ou en minuscules).

Le programme affichera sur la sortie standard :

- « Not Found » si le mot n'a pas été trouvé ;
- (x0,y0)(x1,y1) si le mot a été trouvé avec :
  - (x0,y0) = Abscisse et ordonnée en caractères de la première lettre du mot dans la grille ;
  - (x1,y1) = Abscisse et ordonnée en caractères de la dernière lettre du mot dans la grille.

Les coordonnées (0,0) correspondent au premier caractère de la grille (en haut à gauche).

Par exemple :

```
$ ls
grid solver
$ cat grid
HORIZONTAL
DXRAHCLBGA
DIKCILEOKC
IGAJHYLYHI
HGFGODTIOT
GDLROWKBFR
PLNRDNERGE
JHAIDUAJGV
UKGFFOLLEH
$ ./solver grid horizontal
(0,0)(9,0)
$ ./solver grid vertical
(9,7)(9,0)
$ ./solver grid diagonal
(0,1)(7,8)
$ ./solver grid find
(4,8)(1,5)
$ ./solver grid hello
(9,8)(5,8)
$ ./solver grid world
(5,5)(1,5)
$ ./solver grid goldorak
(8,1)(1,8)
$ ./solver grid epita
Not found
```

2. Exemples d’images

Vous trouverez ci-dessous quelques images qui pourront être utilisées pour tester votre code. La résolution de ces images est bien plus grande que ce qui s’affiche ci-dessous. Certains lecteurs PDF (comme [Evince](#)) vous permettent de sauvegarder ces images dans différents formats.

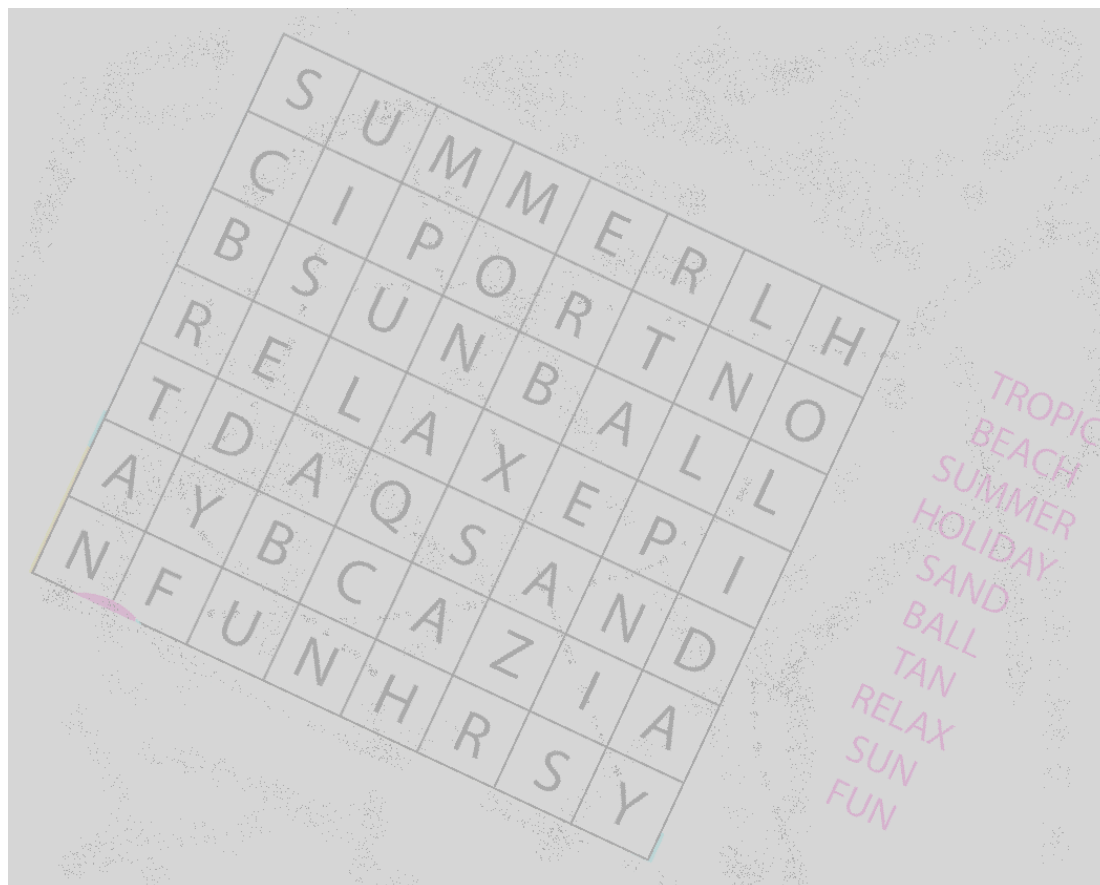
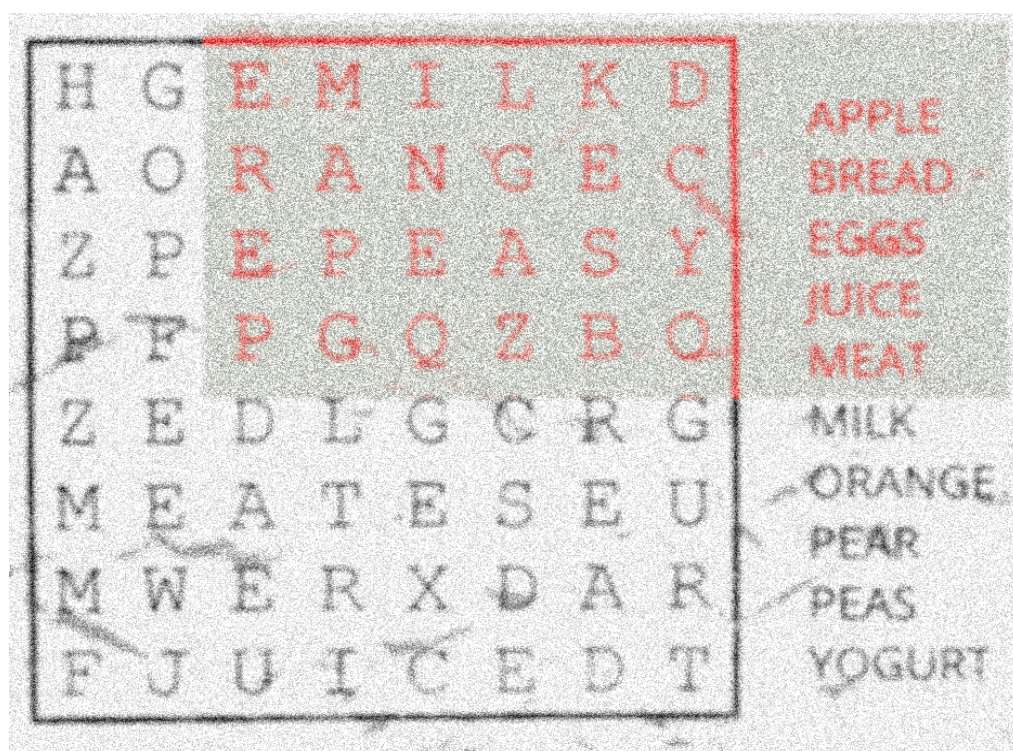
2.1. Niveau 1

M	S	W	A	T	E	R	M	E	L	O	N	APPLE
Y	T	B	N	E	P	E	W	R	M	A	E	LEMON
R	R	L	W	P	A	P	A	Y	A	N	A	BANANA
R	A	N	L	E	M	O	N	A	N	E	P	LIME
E	W	L	E	A	P	R	I	A	B	P	R	ORANGE
B	B	I	L	B	B	W	B	R	L	A	Y	WATERMELON
K	E	M	P	M	A	W	L	R	A	R	B	GRAPE
C	R	E	P	R	N	R	E	R	R	G	R	KIWI
A	R	Y	A	Y	A	O	A	N	L	A	M	STRAWBERRY
L	Y	Y	A	R	N	E	R	K	I	W	I	PAPAYA
B	E	B	A	A	A	N	A	A	P	R	T	BLUEBERRY
Y	R	R	E	B	P	S	A	R	N	N	W	BLACKBERRY
Y	R	R	E	B	E	U	L	B	L	G	I	RASPBERRY
T	Y	P	A	T	E	A	E	P	A	C	E	

MINDFULNESS  
IMAGINE  
RELAX  
COOL  
RESTING  
BREATHE  
EASY  
TENSION  
STRESS  
CALM

P X U T S I N I U P R V G B M D D  
E H A A S P O J P E T B E Q Z L C  
A U N T E G Q T L H R Z F A T O P  
S H X F N G U A X E A A Y P O M H  
Y O Y Y L D X L A K Y U Z L B S K  
J X M U U G Q T R I M A G I N E B  
H F N W F X H D P B B B T N V S K  
H I I H D E S Q F U M Y E R N S X  
R P B Z N H S D S L H O N B S S S  
E H X A I Z I H A H O E S Q F E F  
C W Z I M V D C J V S S I M G R W  
L A I I R Z Q Q H X D Z O Z Q T R  
W C A X E Z R G H A I Z N E C S E  
B R H F O T G N I T S E R E O V Z  
M W V W Q D U I H W Q T S B I M L  
T D T O N Z C X X R G E L K H F Q  
Q N E K S V M O T F A L A A E W B

## 2.2. Niveau 2



2.3. Niveau 3

NAME: \_\_\_\_\_

DATE: \_\_\_\_\_

Strange Words

WORD FIND



Y I M Z W J C E T A V I T S E R J K M X O H Y  
P A L I M P S E S T U X D T T E G C N D M K Y  
R B G N O I T A L U B A N N I T N I T E P D P  
O O Q I G N I K N U L E P S M E D F V T H E U  
P P A N G L O S S I A N Z D C M I T R A A F S  
R Y K J P E T R I C H O R N F O U N E L L E I  
I H F R I P P E T Q J A N C T N V A T U O N L  
O G U F S U S U R R U S X J A A J G X B S E L  
C T A T T E R D E M A L I O N M S A O O K S A  
E D E M O R D N I L A P U N O O T M Y B E T N  
P J R C N X D W E H G N A P S M M R Y M P R I  
T S X M L P E D I A N S W H U G E E G O S A M  
I G N I T A V R E N E J C L M Y S T Y C I T O  
O G E R Y T H R I S M A L I H H I X Z S S E U  
N R C F M O H F G N Y N A L T P S C Y I G P S  
R G G A I S E N M O T P Y R C S C E S D O H C

TINTINNABULATION

DEFENESTRATE

TERMAGANT

DISCOMBOBULATED

PANGLOSSIAN

SUSURRUS

OMPHALOSKEPSIS

ERYTHRISMAL

ESTIVATE

PROPRIOCEPTION

PALINDROME

SPANGHEW

TATTERDEMALION

ENERVATING

FRIPPET

PUSILLANIMOUS

PALIMPEST

SYZYGY

CRYPTOMNESIA

SPELUNKING

TMESIS

Find the words below and circle them (across, down, or diagonal):

☐ FLAMINGO

☐ TOUCAN

☐ SWAN

☐ KIWI

☐ PELICAN

☐ EAGLE

☐ TURKEY

☐ PEACOCK

☐ CARDINAL

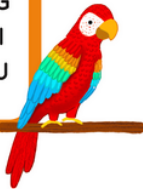
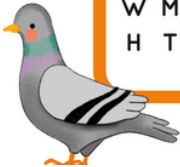
☐ PIGEON

☐ PARROT

☐ SEAGULL



Y O T P E S O P C C X W O M L  
E U R E B H U L I T O R A D O  
A F Z A U G Q R Q G N F D X O  
G N L C B R C I Z R E F K M Y  
L M Y O F L A M I N G O F W L  
E Z V C G L R W D F T P N L M  
T U R K E Y D Y Q F G E U K G  
R J S P E L I C A N Z G V O Y  
T K P W K L N G T Q A U B D L  
D O Y A A M A S X E P X C K X  
K N U W R N L L S O K K Q Q A  
Y M L C D R U W F O T B I M A  
A L K E A A O M Q S P C N W G  
W M G Q I N M T K T B D D R I  
H T V L X M F S O M Y Z I K J






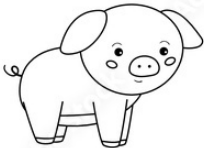


2.4. Niveau 4

**FARM**


W	C	I	Z	G	U	X	D	B	N	R	E	L	S
Q	X	P	V	O	D	H	A	Y	I	M	T	F	K
G	N	L	F	A	R	M	J	K	C	S	Z	W	E
H	V	O	D	T	I	B	S	L	U	E	D	O	G
A	R	W	C	K	H	E	N	F	Y	E	P	X	M
B	S	H	E	E	P	O	M	A	J	D	Y	I	T
Q	Z	U	N	G	V	K	G	R	A	S	S	N	L
F	D	T	R	G	W	C	I	M	P	O	J	U	B
O	A	K	Y	S	J	A	Z	E	H	T	Q	P	V
O	M	Q	X	L	F	T	U	R	L	A	N	I	C
D	L	J	B	A	R	N	P	S	E	M	Y	G	R
T	E	G	M	Q	Z	L	C	O	W	S	K	A	H



**HORSE**


**PIG**

**GOAT**



**CHICK**

**DUCK**

**SHEEP**

**COW**

**DOG**

**CAT**

G	O	A	T	P	B	N	C
J	K	I	S	Z	R	M	A
E	S	C	O	D	W	H	T
D	H	L	H	P	D	H	O
I	E	V	D	I	C	O	W
H	E	Z	O	O	H	R	G
K	P	I	G	T	I	S	R
W	U	X	Q	C	H	E	A
D	U	C	K	F	Y	M	O
G	P	S	C	H	I	C	K

Adobe Stock | #547514068

## 2.5. Liens de téléchargement

Au besoin, ces images peuvent être téléchargées au format PNG à l'aide des liens ci-dessous :

- [http://www.debug-pro.com/epita/prog/s3/project/level\\_1\\_image\\_1.png](http://www.debug-pro.com/epita/prog/s3/project/level_1_image_1.png)
- [http://www.debug-pro.com/epita/prog/s3/project/level\\_1\\_image\\_2.png](http://www.debug-pro.com/epita/prog/s3/project/level_1_image_2.png)
- [http://www.debug-pro.com/epita/prog/s3/project/level\\_2\\_image\\_1.png](http://www.debug-pro.com/epita/prog/s3/project/level_2_image_1.png)
- [http://www.debug-pro.com/epita/prog/s3/project/level\\_2\\_image\\_2.png](http://www.debug-pro.com/epita/prog/s3/project/level_2_image_2.png)
- [http://www.debug-pro.com/epita/prog/s3/project/level\\_3\\_image\\_1.png](http://www.debug-pro.com/epita/prog/s3/project/level_3_image_1.png)
- [http://www.debug-pro.com/epita/prog/s3/project/level\\_3\\_image\\_2.png](http://www.debug-pro.com/epita/prog/s3/project/level_3_image_2.png)
- [http://www.debug-pro.com/epita/prog/s3/project/level\\_4\\_image\\_1.png](http://www.debug-pro.com/epita/prog/s3/project/level_4_image_1.png)
- [http://www.debug-pro.com/epita/prog/s3/project/level\\_4\\_image\\_2.png](http://www.debug-pro.com/epita/prog/s3/project/level_4_image_2.png)