TRAVAIL PRATIQUE 1 – 20 %

MISE EN SITUATION

Votre petit cousin est fasciné par les formes géométriques. A chaque fois que vous le rencontrez, il vous demande de lui dessiner des carrés, des rectangles, des triangles, des losanges. Quelques fois, il les veut plein, d'autres fois vides, il les veut de tailles différentes, des petits, des gros. Vous décidez d'écrire un programme en C++ qui répondra à tous les caprices de votre petit cousin et qu'enfin vous retrouviez un peu la paix !

CONSIGNES

Dès son lancement le programme offre 4 choix de figures et la possibilité de sortir du programme :

```
1.  Rectangle
2.  Triangle
3.  Carré
4.  Losange
5.  Quitter
Votre Choix -->
```

L'utilisateur doit faire un choix parmi ces options. S'il saisit d'autres lettres ou chiffres le programme lui affiche un message d'erreur et lui offre de nouveau les 5 choix.

Une fois la figure choisie, le programme offre la possibilité d'obtenir des formes pleines ou vides ou de retourner au menu précédent :

```
1.  Forme pleine
2.  Forme vide
3.  Retour au menu précédent
Votre Choix -->
```

Enfin le programme demande la hauteur et la largeur des figures. Dans le cas d'un carré, il faudra juste demander la hauteur, puisque ses côtés sont de largeur identique. Dans le cas d'un losange ou d'un triangle, on demandera uniquement la hauteur. La hauteur devra être impaire dans le cas du losange pour pouvoir tracer convenablement la ligne du milieu.

Pour dessiner le triangle, il y a quatre possibilités :

Position 1	**Position 2**	**Position 3**	**Position 4**
`*`	`*****`	`*****`	`*`
`**`	`*###*`	`*###*`	`**`
`*#*`	`*#*`	`*#*`	`*#*`
`*###*`	`**`	`**`	`*###*`
`*****`	`*`	`*`	`*****`

Le programme génère un nombre aléatoire compris entre 1 et 4 pour choisir de dessiner l'une des quatre possibilités.

Le programme confirme les choix demandés en affichant une phrase décrivant la forme qu'il doit dessiner. Par exemple : `Voici votre rectangle vide de 4x25:`

Une fois la forme dessinée à l'écran, le programme affiche de nouveau le menu après avoir effacé l'écran.

EXEMPLE D'EXÉCUTION :

```
Choisissez la forme :
 1.   Rectangle
 2.   Triangle
 3.   Carré
 4.   Losange
 5.   Quitter
Votre choix → 1
Choisissez le remplissage :
 1.   Forme pleine
 2.   Forme vide
 3.   Retour au menu précédent
Votre choix → 2
Entrez la hauteur du rectangle : 4
Entrer la largeur du rectangle : 25
Voici votre rectangle vide de 4x25 :
*************************
*                       *
*                       *
*************************
Appuyez sur une touche pour continuer
Choisissez la forme :
 1.   Rectangle
 2.   Triangle
 3.   Carré
 4.   Losange
 5.   Quitter
Votre choix → 4
Choisissez le remplissage :
 1.   Forme pleine
 2.   Forme vide
 3.   Retour au menu précédent
Votre choix → 1
Entrez la hauteur du losange : 9
Voici votre losange plein de hauteur 9 :
     *
    *#*
   *###*
  *#####*
 *#######*
  *#####*
   *###*
    *#*
     *
Appuyez sur une touche pour continuer
Choisissez la forme :
 1.   Rectangle
 2.   Triangle
 3.   Carré
 4.   Losange
 5.   Quitter
Votre choix → 2
Choisissez le remplissage :
 1.   Forme pleine
 2.   Forme vide
 3.   Retour au menu précédent
Votre choix →  1
Entrez la hauteur du triangle : 5
```

```
Voici votre triangle plein de hauteur 5 :
*
**
*#*
*##*
*****
Appuyez sur une touche pour continuer
Choisissez la forme :
  1.    Rectangle
  2.    Triangle
  3.    Carré
  4.    Losange
  5.    Quitter
Votre choix → 5
```

CONTRAINTES :

Votre programme ne doit utiliser que les notions vues jusqu'à présent (`if`, `while`, `for`, `switch`, `opérateur conditionnel`, `||` et `&&`, `system("cls")`, `system("pause")`, `string`) et les fonctions. N'oubliez pas les commentaires pertinents.

Le programme principal vous est fourni et vous ne devez pas le modifier, à moins d'implémenter des défis. Dans ce cas, seules les parties affectées par les défis pourront être modifiées.

Le contour et le remplissage des formes sont des caractères différents. Par défaut, l'étoile (*) permettra de tracer le contour et le dièse (#) remplira la forme si elle est choisie pleine.

Les formes étant des formes en deux dimensions, le code pour afficher ces formes aura recours **uniquement à deux boucles**, une pour afficher les lignes, une deuxième pour afficher les colonnes.

Vous faites régulièrement des sauvegardes avec l'outil git et vous les téléchargez dans GitHub.

En utilisant la compilation séparée, le programme doit **minimalement** comporter les fonctions suivantes :

`afficherMenu1()` :	Fonction qui affiche le menu 1
`afficherMenu2()` :	Fonction qui affiche le menu 2
`validerMenu()` :	Fonction qui lit le choix de l'utilisateur et vérifie que le choix est bien parmi les choix offerts dans le menu, informations passées en paramètre. Cette fonction retourne le choix validé. L'utilisateur peut taper n'importe quoi au clavier, le programme ne part pas en boucle infinie.
`traiter`***`Forme`***`()` :	Ces fonctions, selon la forme, demandent les dimensions de la forme, affichent la phrase de présentation de la forme avec les bonnes dimensions et finalement appellent la fonction dessinant la forme. Il y aura donc 4 versions de cette fonction : `traiterRectangle()`, `traiterCarre()`, `traiterTriangle()`, `traiterLosange()`.
`dessinerRectangle()` :	Fonction qui dessine un rectangle ou un carré dont la hauteur, la largeur et le mode de remplissage sont passés en paramètre.
`dessinerTriangle1()` :	Fonction qui dessine un triangle dans la position 1 dont la hauteur et le mode de remplissage sont passés en paramètre.
`dessinerTriangle2()` :	Fonction qui dessine un triangle dans la position 2 dont la hauteur et le mode de remplissage sont passés en paramètre.
`dessinerTriangle3()` :	Fonction qui dessine un triangle dans la position 3 dont la hauteur et le mode de remplissage sont passés en paramètre.
`dessinerTriangle4()` :	Fonction qui dessine un triangle dans la position 4 dont la hauteur et le mode de remplissage sont passés en paramètre.

`dessinerLosange()` : Fonction qui dessine un losange dont la hauteur et le mode de remplissage sont passés en paramètre.

`genererNombreAleatoire()` : Fonction qui retourne un nombre aléatoire compris entre min et max passés en paramètre.

DÉFIS :

Vous pouvez, si le cœur vous en dit,

- proposer d'autres formes, comme des triangles isocèles, des cercles, des hexagones, des octogones,… (1 point bonis par forme additionnelle)
- offrir le choix du motif permettant de dessiner le contour de la forme et un autre pour le remplissage de la forme, (1 point bonis)
- centrer la figure dans le milieu de l'écran, (1 point bonis)
- vous pouvez également créer des fonctions supplémentaires, comme, (1 point bonis par fonction pertinente)
  - `lireDimensionValide` : fonction qui retourne une dimension comprise entre un minimum (par exemple 3 pour que la forme ressemble à quelque chose) et un maximum (24 est nombre de ligne maximum de l'écran ou 79 est nombre de caractères maximum par ligne).
  - `positionnerCurseur` : fonction qui place le curseur à la coordonnée x et y dans l'écran. Cela permettra par la suite de dessiner la forme à partir de x et y.
- laissez aller votre imagination !!!

LIVRABLE :

- Vous devez créer une solution **SolutionTPPrenomNom** (par exemple : SolutionTpKarineMoreau) qui contiendra un projet **Formes** suivi de **votre Matricule** (par exemple Formes1234567). Les fichiers sources seront enregistrés avec **votre matricule**, par exemple `Fonctions1234567.cpp`, `Fonctions1234567.h`. `Main1234567.cpp`. Le non-respect de cette consigne pourra entraîner une pénalité de 5%.
- Vous déposez le dossier compressé sans le dossier .vs de la solution dans **LEA avant** le début du cours du **17 novembre 2020**. Toute remise pendant le cours entraînera un retard de -10 %.
- Vous rendez publique votre solution dans GitHub et vous envoyez le lien au professeur via outlook avec un message ayant comme objet : TP - Les fonctions

ÉVALUATION :

• Toutes les variables et constantes sont correctement définies	5%
• Le programme est découpé en fonctions et les fonctions sont définies dans les fichiers séparés. Les prototypes des fonctions sont correctement définis	10%
• Le code source est largement commenté et les commentaires sont pertinents	20%
• La structure des fonctions est logique et pertinente	25%
• Le programme s'exécute conformément aux spécifications	40%

PÉNALITÉS :

- **Livrable :** Solution non conforme : 5%
  Remise non conforme : 5%
- **Orthographe :** maximum 10%
- **Retard :** 10% par jour de retard, après 3 jours de retard, la note zéro sera attribuée

- **Plagiat :** note 0 à toutes les personnes impliquées