Code branche	Ministère de l'Education nationale et de la Formation professionnelle EXAMEN DE FIN D'ETUDES SECONDAIRES TECHNIQUES Régime technique – Division technique générale **Section informatique - Session 2012/2013**	
PROGR D		
Épreuve **écrite**	*Branche*	*Division / Section*
Durée épreuve **4 h**	**Programmation - Delphi**	**GI** **Technique générale** **Informatique**
Date épreuve 17 mai 2013		

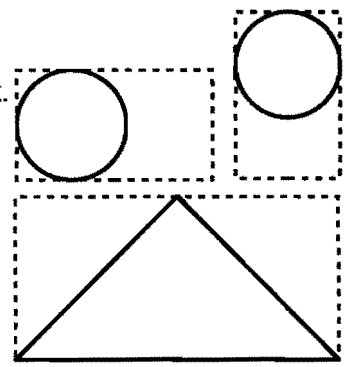
*Dans votre répertoire de travail, vous trouverez un sous-dossier nommé **EXAMEN_GI_xx**. Renommez ce dossier en remplaçant le nom par votre code de l'examen (Exemple de notation : **LTAM_GI_07**). Tous vos fichiers devront être sauvegardés à l'intérieur de ce sous-dossier, qui sera appelé 'votre dossier' dans la suite !*

Question 1 - Liste chaînée de figures [22 points]

*Ouvrez le projet **FiguresList.dpr** qui se trouve dans votre dossier **Q1**. Vous allez compléter ce projet comme décrit dans la suite. Le programme de démonstration **FigureList.exe** se trouve dans votre dossier. L'unité **UMain.pas** est donnée et complète. Les parties qui utilisent les classes que vous devrez définir sont mises en commentaires. Pour tester, vous devrez enlever les marques des commentaires au fur et à mesure que vous avancez.*

En profitant au mieux de l'encapsulation, de l'héritage et du polymorphisme, vous allez programmer une **liste chaînée** de figures. Les figures peuvent être de trois types différents : rectangles, cercles et triangles isocèles *(DE : gleichschenklig)*. Pour toutes les figures, on mémorise la couleur et le rectangle circonscrit à la figure :

- Pour les rectangles, la figure correspond au rectangle circonscrit.
- Pour les cercles, la figure est le plus grand cercle que l'on peut dessiner (en haut à gauche) dans le rectangle circonscrit.
- Pour les triangles, la figure est le triangle isocèle dont la base correspond au côté inférieur du rectangle et la pointe supérieure se trouve au milieu du côté supérieur du rectangle circonscrit.



Détails de la réalisation :

Les classes à ajouter s'appellent **TFigureList, TFigure, TRectangle, TTriangle, TCircle.**
Définissez chaque classe dans un fichier individuel et respectez les règles de l'encapsulation (c.-à-d. employez toujours la visibilité la plus restreinte possible).

Les classes **TFigure, TRectangle, TTriangle, TCircle [12 points] :**

Chaque figure est initialisée lors de la construction et elle ne peut plus être changée dans la suite. Chaque figure est définie par les coordonnées de son point supérieur gauche (**X,Y**), sa largeur **Width,** sa hauteur **Height** et sa couleur **Color** (→ voir aussi *pbDrawMouseUp*).
Chaque figure possède les méthodes suivantes :

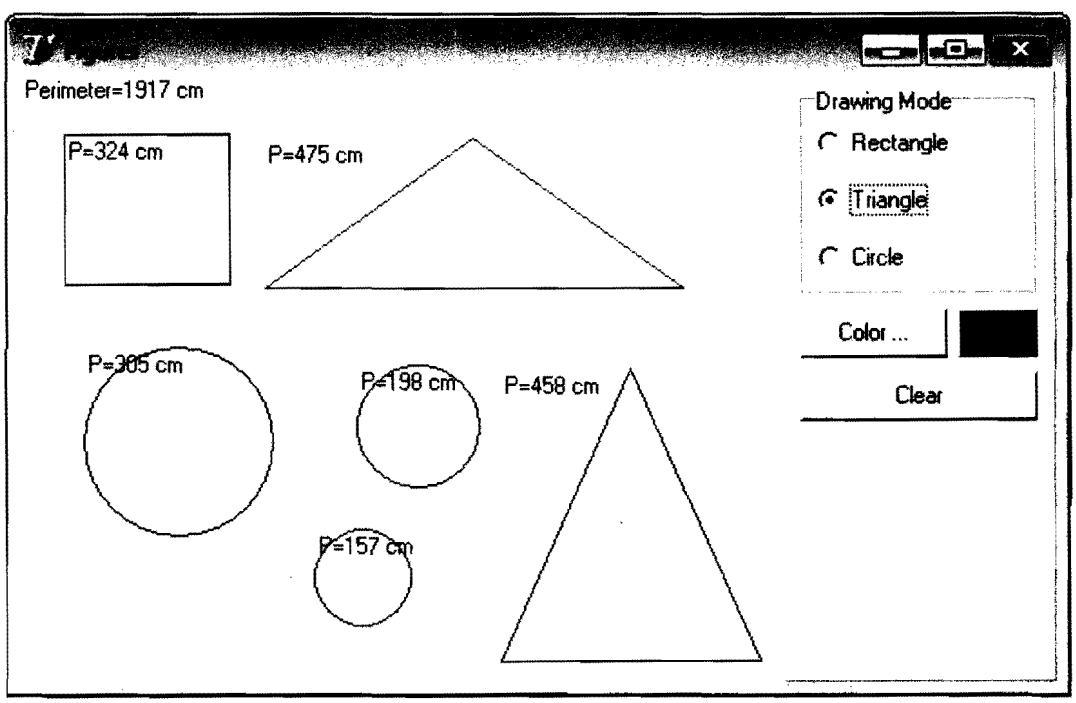
- **CalculatePerimeter** retourne le périmètre de la figure en cm (nombre entier),
- **Draw(pCanvas : TCanvas)** dessine la figure (uniquement la bordure) dans sa couleur ET inscrit son périmètre en bleu en haut à gauche du rectangle circonscrit de la figure (→ voir modèle).

Pour les calculs, on considère que 1 point (pixel) à l'écran correspond à 1 cm en réalité.
Rappel : profitez au mieux de l'héritage et du polymorphisme pour éviter la répétition du même code dans votre programme.

La classe **TFigureList [10 points] :**

La classe **TFigureList** contient et gère la **liste chaînée** des figures. Elle possède les méthodes suivantes :

- **Add(pFigure : TFigure)** ajoute une figure à la fin de la liste chaînée,
- **Clear** efface toutes les figures de la liste et libère la mémoire,
- **CalculatePerimeter** retourne le périmètre total des figures en cm,
- **Draw(pCanvas : TCanvas)** dessine toutes les figures. Le périmètre total est affiché en bleu dans le coin supérieur gauche du canevas (→ voir modèle).



Question 2 - Jeu Bubbles [38 points]

Bubbles est un jeu qui se joue dans une grille rectangulaire qui contient des bulles de différentes couleurs. Il s'agit, de faire 'éclater' des groupes de bulles de même couleur. On appelle ici **'groupe de bulles'**, deux ou plusieurs bulles de même couleur qui se touchent en horizontale ou en verticale. Au moins deux boules de la même couleur doivent se toucher pour qu'on puisse les faire éclater. On fait éclater un groupe de bulles en cliquant sur l'une des bulles du groupe, les autres bulles du groupe sont éclatées récursivement. Après que les bulles ont éclaté, les bulles qui se trouvent au-dessus de celles qui ont disparu 'tombent' vers le bas, de façon à ce que les bulles restent toujours tassées en verticale.

*Ouvrez le programme de démonstration **Bubbles.exe** qui se trouve dans votre dossier pour vous familiariser avec le fonctionnement du jeu !*

*Créez ensuite un nouveau projet **Bubbles.dpr** dans le sous-dossier **Q2** de votre dossier. Réalisez le jeu comme décrit dans la suite.*

La classe TBubbles (unit UBubbles.pas) [30 points] :

Les bulles dans le jeu peuvent avoir 5 couleurs différentes : jaune, rouge, vert, bleu, orange.

Définissez d'abord le type **TBubble** qui sert à définir les bulles qui se trouvent dans la grille :

TBubble = (bbNone, bbYellow, bbRed, bbGreen, bbBlue, bbOrange)

bbNone indique qu'il ne se trouve pas de bulle à une position donnée.

Dans notre jeu, les bulles sont gérées à l'intérieur de la classe **TBubbles** dans un tableau privé **arBubbles** à deux dimensions. Les éléments de **arBubbles** sont du type **TBubble**. Les dimensions actuelles de **arBubbles** sont mémorisées dans deux attributs privés **Cols** et **Rows.** Le jeu est limité à un maximum de 100 bulles par ligne et par colonne.

Dans notre programme, le canevas du jeu peut être redimensionné. Les bulles doivent cependant être des cercles (pas des ellipses) et la grille des bulles doit occuper l'espace maximal possible sur le canevas. Ainsi, le diamètre pour les bulles doit être recalculé à chaque dessin (→ dans la méthode **Draw**). Pour être disponible à tout moment, le diamètre est mémorisé dans une propriété à lecture seule **Diameter.** [2p.]

Le score est mémorisé dans une propriété à lecture seule **Score**.

- Le constructeur **Create(pCol,pRow : Integer)** a les charges suivantes : [2p.]
  - fixer les dimensions de la grille aux dimensions passées comme paramètre,
  - remettre le score à zéro,
  - initialiser chaque cellule de la grille par une bulle dont la couleur est choisie au hasard.
- La méthode **Draw(pCanvas:TCanvas ; pWidth,pHeight:Integer)** a les charges suivantes : [5p.]
  - recalculer le diamètre,
  - dessiner le fond de la grille (gris foncé) et toutes les bulles. La bordure et l'intérieur des bulles ont la même couleur.

- La méthode **Pop(pCol,pRow,pPoints : integer)** a les charges suivantes : [10p.]
  - supprimer la bulle aux coordonnées (pCol,pRow) si elle fait partie d'un groupe de bulles de même couleur et supprimer ensuite **récursivement** toutes les bulles connectées horizontalement ou verticalement. Une bulle est 'supprimée' en affectant **bbNone** à sa position dans la grille. La méthode **Pop** ne doit pas tasser les bulles.
  - actualiser le score : le nombre de points initial est 10, ensuite, à chaque niveau de récursivité, le nombre de points par bulle augmente de 10 points, c.-à-d. :

    - 1ère bulle (la bulle cliquée) : 10 points,
    - bulle(s) éliminée(s) par la première bulle : 20 points,
    - bulle(s) éliminée(s) par une bulle à 20 points : 30 points,
    - bulle(s) éliminée(s) par une bulle à 30 points : 40 points,
    - etc.

  - Conseil : Définir 4 variables booléennes **popUp**, **popDown**, **popLeft**, **popRight** qui sont **true**, s'il se trouve une bulle de même couleur respectivement au-dessus, en-dessous, à gauche ou à droite de la bulle actuelle.
- La méthode **Drop** doit être appelée après la suppression d'un groupe de bulles. Elle a la charge de remplir les trous (laissés par les bulles supprimées) par les bulles qui se trouvent au-dessus. C.-à-d. les bulles sont tassées vers le bas (→ voir modèle). [7p.]
- La méthode **SaveToFile(FileName:string)** sauvegarde l'état actuel de la grille dans un fichier texte. Format du fichier texte : [4p.]
  - 1ère ligne : le nombre de colonnes de la grille
  - 2e ligne : le nombre de lignes de la grille
  - 3e ligne : le score actuel du jeu
  - ensuite, le fichier contient autant de lignes qu'il y a de lignes dans la grille. Chaque ligne du fichier contient autant de chiffres qu'il y a de colonnes dans la grille. Utilisez comme codes (chiffres) pour les bulles leur ordre dans le type d'énumération **TBubble**, c-à-d. :

    0:[vide], **1**:jaune, **2**:rouge, **3**:vert, **4**:bleu, **5**:orange

La classe frmMain (unit UMain.pas) [8 points] :

- Réalisez le programme en prenant comme référence le programme de démonstration **Bubbles.exe**.
- Le bouton *'NewGame'* crée une nouvelle instance de **TBubbles**. Ignorez le clic sur *'New Game'* si l'un des deux champs d'édition contient des données incorrectes.
- Lors de la fermeture du programme, l'état actuel du jeu est mémorisé dans le fichier **Bubbles.txt** qui est à localiser dans le dossier du projet.
- Evitez les erreurs qui peuvent se produire après le démarrage du programme si on n'a pas encore créé une instance de **TBubbles**.