Coder une calculatrice en Pascal Objet avec Lazarus

version du 25/03/2019

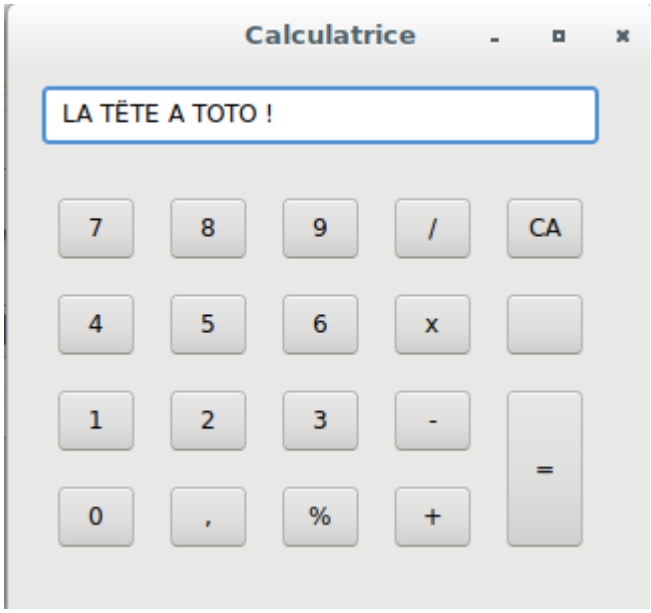
TP : Coder le logiciel « calculatrice »

Nom : **Prénom :** **Classe :** **Date :**	**Appréciation :**	**Note :**
Objectifs : tu devras être capable de coder une calculatrice basique avec l'EDI Lazarus, en suivant la trame fournie.		**durée** : 3h

Matériel

- un ordinateur Windows 10
- le logiciel Lazarus IDE

Travail à réaliser :

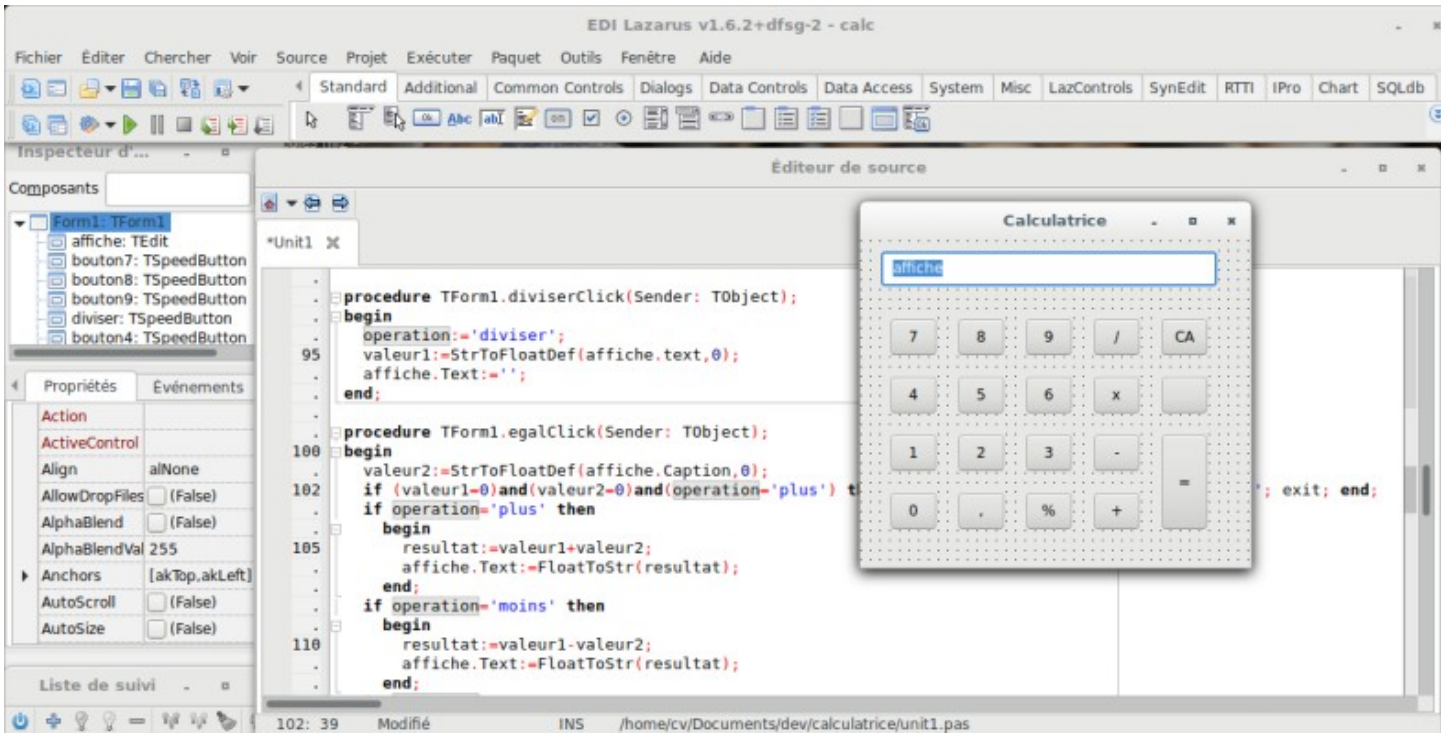


La calculatrice est capable d'effectuer des opérations de base, en réagissant aux clics sur les différents boutons.

Un clic sur le bouton `=` déclenche le calcul du résultat, puis son affichage.

Un clic sur le bouton `CA` déclenche l'effacement de la zone d'affichage.

Préparation du projet



- Télécharge et installe le logiciel Lazarus IDE
- Créer un dossier « calculatrice » dans « Mes documents »
- Lances le logiciel Lazarus IDE
- Créer un nouveau projet (Projet → Nouveau Projet → Application)
- Enregistrer le Projet sous le nom « **calc.lpi** », dans le dossier « calculatrice » dans « Mes documents », puis enregistrer la fiche principale sous le nom « **unit1** » (Projet → Enregistrer le Projet)
- Change le titre de la fiche principale en « **Calculatrice** » (il faut modifier la propriété « **caption** » de la fiche (voir page suivante).

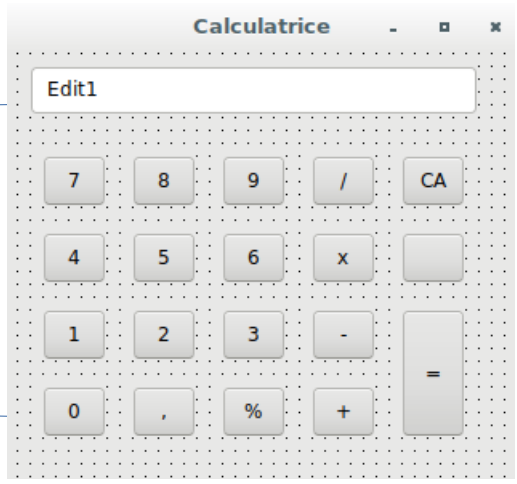
1ère objectif : créer la fiche principale du programme

- Ajouter les éléments suivants à la fiche :

note : le texte des SpeedButton peut être modifié par leur propriété « caption »

1 élément TEdit

19 éléments TSpeedButton

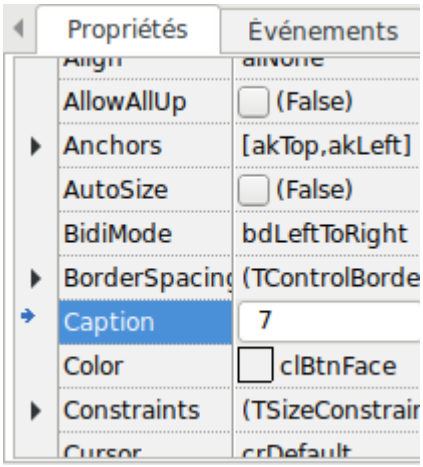


- modifier les propriétés « name » de tous les éléments :

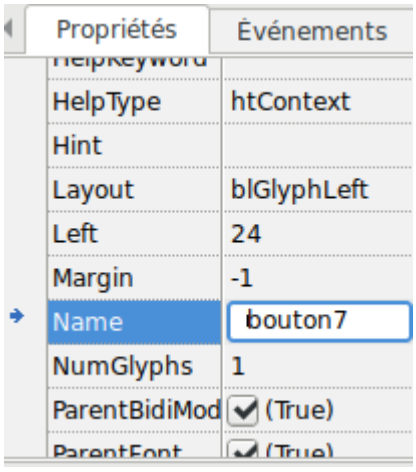
Élément	« name »
Bouton « 0 »	bouton0
Bouton « 1 »	bouton1
...	...
Bouton « 9 »	bouton9
Bouton « diviser »	diviser
Bouton « multiplier »	multiplier

Élément	« name »
Bouton « moins »	moins
Bouton « plus »	plus
Bouton « virgule »	virgule
Bouton «pourcentage»	pourcentage
Bouton « egal »	egal
Bouton « CA »	CA
TEdit	affiche

Aide : propriété Caption

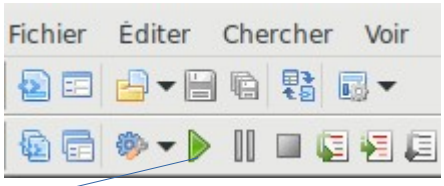


Aide : propriété Name



2ème objectif : gérer les clics sur les boutons

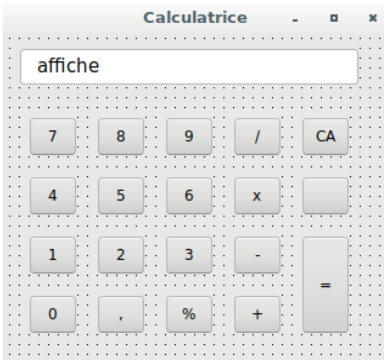
A ce stade, l'application se lance et affiche une calculatrice… mais elle ne fait rien !!



Testes le logiciel
en cliquant sur ce bouton

Nous allons d'abord gérer les clics sur le bouton CA

Qu'est-ce qui doit être modifié quand l'utilisateur clique sur le bouton CA?



la chaîne de caractère dans « affiche » doit être supprimé (chaîne vide)

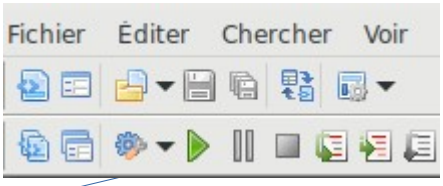
Voici comment procéder :

Double-clic sur le bouton « CA » de la fiche ; le code suivant est ajouté automatiquement :

```
procedure TForm1.CAClick(Sender: TObject);
begin

end;
```

Complète le code de la façon suivante :

```
procedure TForm1.CAClick(Sender: TObject);
begin
  affiche.Text:=''; .
end;
```



Testes le bouton CA
en cliquant sur ce bouton

L'affichage est-il « effacé » quand tu cliques sur le bouton « CA » ?

Nous allons maintenant gérer les clics sur les chiffres de 0 à 9

Qu'est-ce qui doit être modifié quand l'utilisateur clique sur un des chiffres ?

Le chiffre doit être rajouté à droite de la chaîne de caractère dans « affiche »

Voici comment procéder :

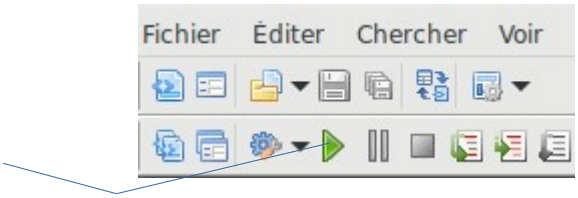
Double-clic sur le chiffre « 0 » de la fiche ; le code suivant est ajouté automatiquement :

```
procedure TForm1.bouton0Click(Sender: TObject);
begin

end;
```

Complète le code de la façon suivante :

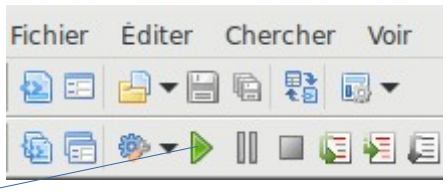
```
procedure TForm1.bouton0Click(Sender: TObject);
begin
  affiche.Text:=affiche.Text + '0';
end;
```

Testes le bouton CA
en cliquant sur ce bouton



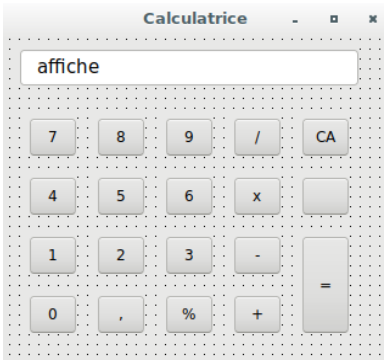
Le chiffre « 0 » est-il ajouté à droite dans la chaîne « affiche » quand tu cliques sur le bouton « 0 » ?

Testes le bouton CA
en cliquant sur ce bouton



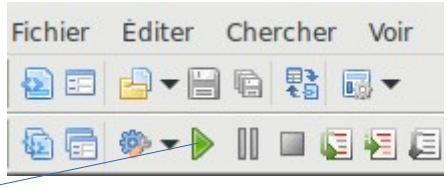
Fais de même pour gérer les clics sur les chiffres :

« 1 », « 2 », « 3 », « 4 », « 5 », « 6 », « 7 », « 8 », « 9 » et « , »



en adaptant le code utilisé pour le chiffre « 0 »

Testes les chiffres de 1 à 9 en cliquant sur ce bouton



A ce stade tu es capable de rentrer n'importe quel nombre dans la calculatrice

Avant de commencer à programmer la réponse aux clics sur les boutons « + », « - », « x » et « / », tu dois comprendre 2 notions importantes :

- les variables
- la programmation événementielle

Les variables

Le variables servent à garder en mémoire des valeurs importantes pour le fonctionnement du logiciel.
Le tableau ci-dessous donne une liste de quelques-unes des variables de notre logiciel « calculatrice ».
Les variables peuvent être de type entier (***int***), réels (***real***), chaînes de caractère (***string***), etc...

Nom de la variable	type	signification
valeur1	Real	1er nombre de l'opération
valeur2	Real	2ème nombre de l'opération
resultat	Real	Le résultat de l'opération
operation	String	mémorise le type d'opération choisi : "plus","moins","multiplier","diviser","%"

Ces variables doivent être déclarée au début du programme, dans la section **var** :

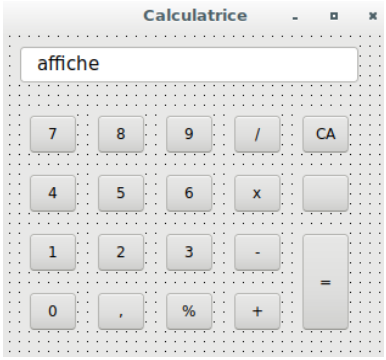
```
var
  Form1: TForm1;
  operation : String;
  valeur1 : Real;
  valeur2 : Real;
  resultat : Real;

implementation
```

La programmation événementielle

Les fonctions du programme sont « déclenchées » par des évènements :

- ✔ clic de souris
- ✔ appui sur le clavier
- ✔ etc...

Nous allons maintenant gérer les clics sur le bouton « + »



Qu'est-ce qui doit être modifié quand l'utilisateur clique sur ce bouton ?

- La valeur « plus » doit être affectée à la variable « operation »

- la valeur affichée dans « affiche » doit être affectée à la variable « valeur1 » sous formé d'un nombre réel

- « affiche » doit être effacé pour permettre la saisie du 2ème nombre de l'opération

Voici comment procéder :

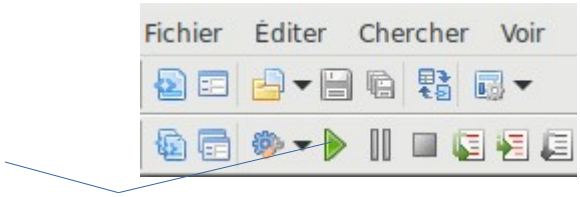
Double-clic sur le bouton « + » de la fiche ; le code suivant est ajouté automatiquement :

```
procedure TForm1.plusClick(Sender: TObject);
begin

end;
```

Complète le code de la façon suivante :

```
procedure TForm1.plusClick (Sender: TObject);
begin
  operation:='plus';
  valeur1:=StrToFloatDef(affiche.text,0);
  affiche.Text:='';
end;
```

Testes l'effet du bouton + en cliquant sur ce bouton

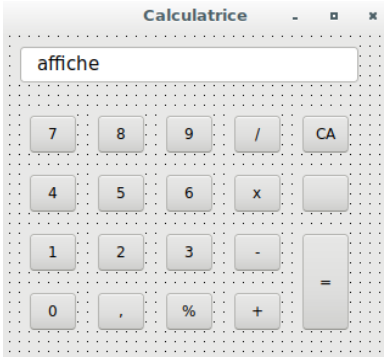


la chaîne « affiche » est-elle effacée quand tu cliques sur le bouton « + » ?

la variable « operation » a-t-elle pris la valeur « plus » ?

la valeur correspondant au nombre affiché au moment de l'appui sur le bouton a-t-elle été affectée à valeur1 ?

Nous allons maintenant gérer les clics sur le bouton « - »



Qu'est-ce qui doit être modifié quand l'utilisateur clique sur ce bouton ?

- La valeur « moins » doit être affecté à la variable « operation »

- la valeur affichée dans « affiche » doit être affectée à la variable « valeur1 » sous formé d'un nombre réel

- « affiche » doit être effacé pour permettre la saisie du 2ème nombre de l'opération

Voici comment procéder :

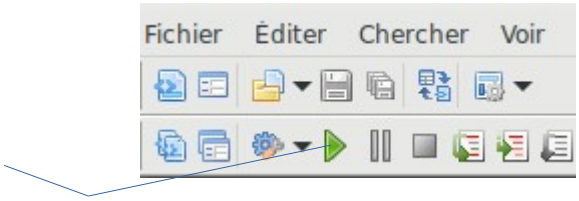
Double-clic sur le bouton « - » de la fiche ; le code suivant est ajouté automatiquement :

```
procedure TForm1.moinsClick(Sender: TObject);
begin

end;
```

Complète le code de la façon suivante :

```
procedure TForm1.moinsClick (Sender: TObject);
begin
  operation:='moins';
  valeur1:=StrToFloatDef(affiche.text,0);
  affiche.Text:='';
end;
```

Testes l'effet du bouton « - » en cliquant sur ce bouton

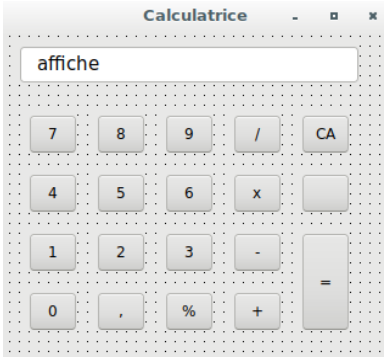


la chaîne « affiche » est-elle effacée quand tu cliques sur le bouton « - » ?

la variable « operation » a-t-elle pris la valeur « moins » ?

la valeur correspondant au nombre affiché au moment de l'appui sur le bouton a-t-elle été affectée à valeur1 ?

Fais de même gérer les clics sur le bouton « x »



Qu'est-ce qui doit être modifié quand l'utilisateur clique sur ce bouton ?

- Le type d'opération « multiplier » doit être entré dans la variable « operation »

- la valeur affichée dans « affiche » doit être entrée dans la variable « valeur1 » sous formé d'un nombre réel

- « affiche » doit être effacé pour permettre la saisie du 2ème nombre de l'opération

Voici comment procéder :

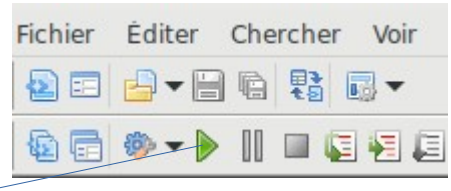
Double-clic sur le bouton « x » de la fiche ; le code suivant est ajouté automatiquement :

```
procedure TForm1.multiplierClick(Sender: TObject);
begin

end;
```

Complète le code :

```
procedure TForm1.multiplierClick(Sender: TObject);
begin



end;
```

Testes l'effet du bouton « x » en cliquant sur ce bouton

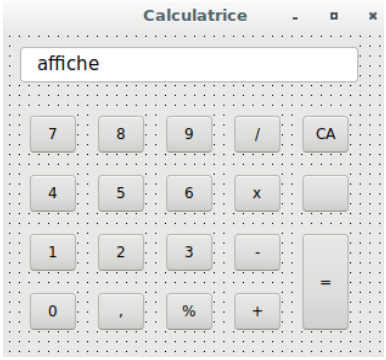


la chaîne « affiche » est-elle effacée quand tu cliques sur le bouton « x » ?

la variable « operation » a-t-elle pris la valeur « multiplier » ?

la valeur correspondant au nombre affiché au moment de l'appui sur le bouton a-t-elle été affectée à valeur1 ?

Fais de même gérer les clics sur le bouton « / »



Qu'est-ce qui doit être modifié quand l'utilisateur clique sur ce bouton ?

- Le type d'opération « diviser » doit être entré dans la variable « operation »

- la valeur affichée dans « affiche » doit être entrée dans la variable « valeur1 » sous formé d'un nombre réel

- « affiche » doit être effacé pour permettre la saisie du 2ème nombre de l'opération

Voici comment procéder :

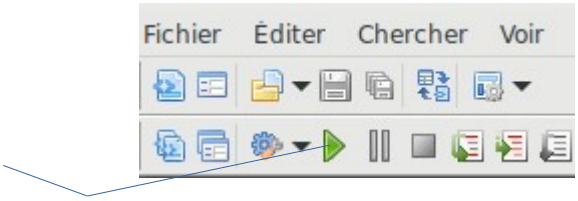
Double-clic sur le bouton « / » de la fiche ; le code suivant est ajouté automatiquement :

```
procedure TForm1.diviserClick(Sender: TObject);
begin

end;
```

Complète le code :

```
procedure TForm1.diviserClick(Sender: TObject);
begin



end;
```

Testes l'effet du bouton « / » en cliquant sur ce bouton

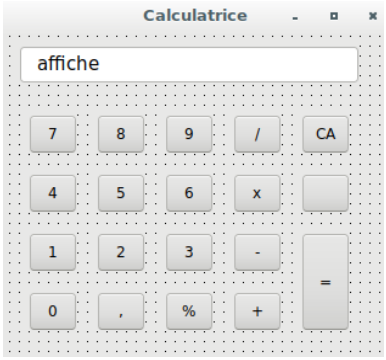


la chaîne « affiche » est-elle effacée quand tu cliques sur le bouton « / » ?

la variable « operation » a-t-elle pris la valeur « diviser » ?

la valeur correspondant au nombre affiché au moment de l'appui sur le bouton a-t-elle été affectée à valeur1 ?

Nous allons maintenant gérer les clics sur le bouton « = »



Qu'est-ce qui doit être modifié quand l'utilisateur clique sur ce bouton ?

- la valeur affichée dans « affiche » doit être entrée dans la variable « valeur2 » sous formé d'un nombre réel

- l'opération doit être effectuée entre valeur1 et valeur2 en tenant compte de la variable « operation » ; le résultat sera affecté à la variable « resultat »

- le résultat de l'opération doit être affiché dans « affiche »

- la variable « operation » doit être réinitialisée

Voici comment procéder :

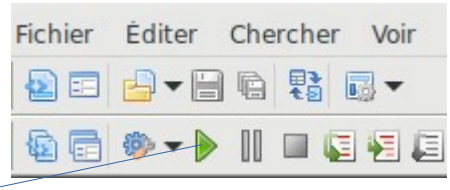
Double-clic sur le bouton « = » de la fiche ; le code suivant est ajouté automatiquement :

```
procedure TForm1.egalClick(Sender: TObject);
begin

end;
```

Complète le code, en remplissant aussi les lignes vides :
*(fais une recherche internet pour découvrir la syntaxe des opérations **moins**, **multiplier** et **diviser**)*

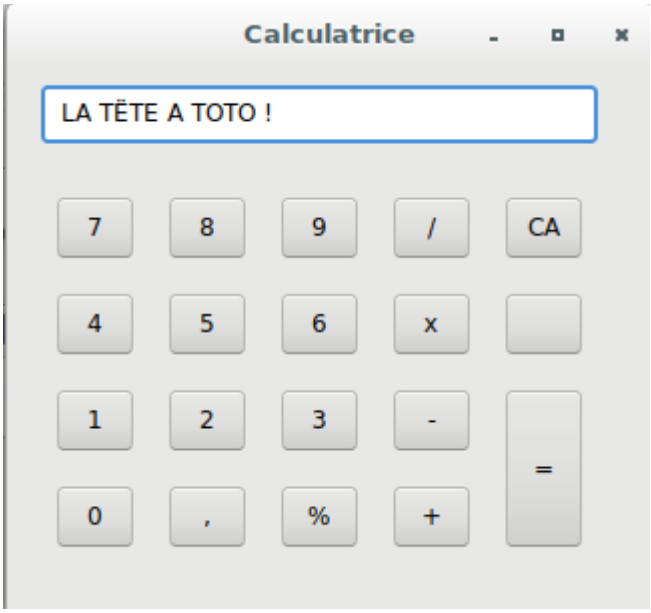
```
procedure TForm1. egalClick(Sender: TObject);
begin
  valeur2:=StrToFloatDef(affiche.Caption,0);
  if operation='plus' then
    begin
      resultat:=valeur1 + valeur2;
      affiche.Text:=FloatToStr(resultat);
    end;
  if operation='moins' then
    begin
      resultat:=                    ;
      affiche.Text:=FloatToStr(resultat);
    end;
  if operation='multiplier' then
    begin
      resultat:=                    ;
      affiche.Text:=FloatToStr(resultat);
    end;
  if operation='diviser' then
    begin
      resultat:=                    ;
      affiche.Text:=FloatToStr(resultat);
    end;
  operation:='';
end;
```

Testes l'effet du bouton « - » en cliquant sur ce bouton



la chaîne « affiche » affiche-t-elle le résultat de l'opération ?

BONUS : intégrer un « easter egg » dans le logiciel



Ajoute une ligne de code dans la fonction **TForm1. EgalClick**, pour que :

- ✔ Quand l'utilisateur appuie sur « **=** » après avoir entré l'opération « **0 + 0** », l'afficheur affiche : « **la tête à Toto !** »

indice : il y a 3 conditions à remplir : valeur1=0 **ET** valeur2=0 **ET** operation='plus' ; tu dois traduire cela en langage informatique.