## *Introduction*

Dans le présent chapitre nous allons invoqués Les algorithmes **d'approximation** ou nous allons les utiliser afin de trouver des valeurs approchées pour des solutions à certains problèmes qui n'ont pas de valeurs exactes comme solution.

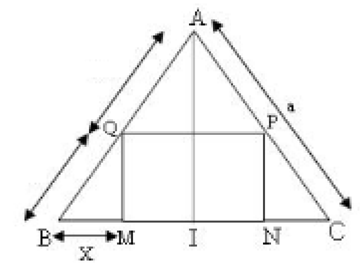
- Un algorithme **d'approximation** ne peut pas trouver une solution optimale mais une valeur approchée da la solution.
- Un algorithme **d'optimisation** est un algorithme qui trouve une solution **optimale** (la meilleure solution) parmi un ensemble de solutions réalisables à un problème de minimisation ou de maximisation.

## I. *Les algorithmes d'optimisations*

**Activité 1**

Soit un triangle équilatéral dont le coté mesure **a** en cm. On inscrit dans ce triangle un rectangle MNPQ. On pose BM=x

Ecrire un programme intitulé **Optimisation** permettant de déterminer la meilleure valeur de **X afin q**ue l'aire du rectangle soit maximale **« Xop et Amax »**




**NB : On suppose que que la valeur du $10^{-4}$<=pas<=$10^{-1}$ et a =3 cm**

### *Exercice Application N°1*

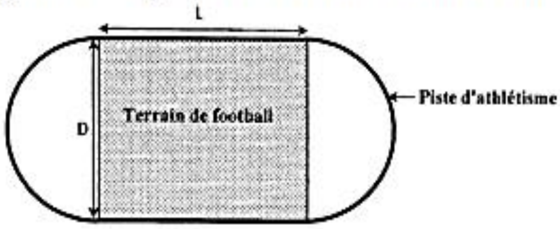
La direction d'une association sportive veut construire un stade formé par une piste d'athlétisme et un terrain de football, tout en cherchant à maximiser la surface de ce dernier.

Le terrain de football est un rectangle de longueur **L**, de largeur **D** et de surface **S**.

La piste d'athlétisme est de longueur **P** et former par les deux arrêts parallèles du terrain de football (de longueur **2*L**) et les deux demi-cercles de diamètre **D** (de longueur $\pi$ * **D**), comme le montre le schéma ci-dessous :

Puisque $S = L * D$ et $P = 2 * L + \pi * D$

Alors $S = L * (P - 2 * L) / \pi$



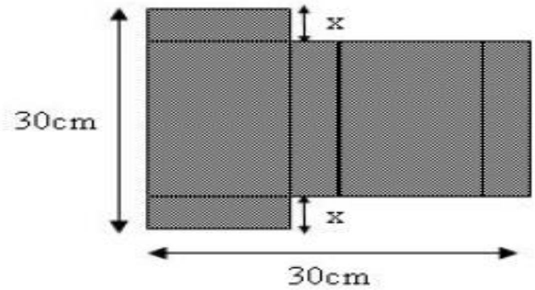

**Travail demandé :**

Etant donné que **L** varie de **0** à **P/2**, écrire un algorithme d'une fonction qui permet de déterminer, à **ε** près, la longueur optimale $L_{opt}$ correspondante à la surface maximale $S_{max}$ du terrain, sachant que **ε** et **P** sont saisis dans l'algorithme du programme principal.

### *Exercice Application N°2*

Un fabricant envisage la production de boites de lait en carton obtenues selon le patron ci-dessous. On se propose d'écrire un module permettant de déterminer la valeur de **xop** pour laquelle le **volume** d'une boite est **maximal vmax**.

NB : **On suppose que que la valeur du $10^{-4}$<=pas<=$10^{-2}$**






## II. Calcul de valeurs approchées de constantes connues

Il existe plusieurs constantes numériques exemple :

- **e (nombre de Neper) ~ 2.718**
- **π (nombre pi) ~ 3.14**
- **etc…**

Dans ce qui suit, nous allons présenter des algorithmes permettant de calculer des valeurs approchées pour les constantes **π** et **e.**

### Activité N°1

On va utiliser les algorithmes d'approximation pour trouver une valeur approchée de la constante e :

**e = 1 + 1/1 + 1/1*2 + 1/1*2*3 + 1/1*2*3*4 + etc..**

On vous demande d'écrire un module permettant de trouver une valeur approchée de la constante **e** à $10^{-4}$ prés

### Activité N°2

Il existe d'autre forme de formule permettant de calculer une valeur approchée de **π on vous demande d**'écrire un module intitulé **Calcul_PI,** permettant de **calculer** et **afficher** une valeur **approchée** de **π** à une valeur **epsilon** donné en paramètre, avec **$10^{-5}$ <=eps<=$10^{-2}$** en utilisant la formule suivante :

$$\frac{\pi}{4} = \frac{3}{4} + \frac{1}{2*3*4} - \frac{1}{4*5*6} + \frac{1}{6*7*8} \cdots$$

### Exercice Application

Soit la formule suivante qui permet de déterminer une valeur approchée de sin(x) sachant que x est un angle mesuré en radian.

$$\sin(x) = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \cdots = \sum_{n=0}^{+\infty} (-1)^n \frac{x^{2n+1}}{(2n+1)!}$$

Ecrire un algorithme d'un module intitulé **calcul_sin** qui permet de :

- Calculer et afficher une valeur approchée de Sin(x) en utilisant la formule donnée ci-dessus pour une valeur de x donnée (déjà saisie au niveau du programme appelant). Le calcul s'arrête lorsque la différence entre deux termes consécutifs devient inférieure ou égale à **$10^{-5}$**.

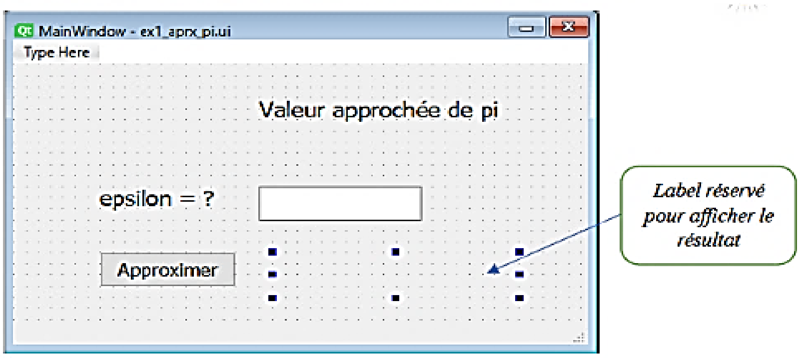
### Exercice N°1

Ecrire un programme modulaire intitulé **Pi_Wallis,** permettant de **calculer** et **afficher** une valeur **approchée** de **π** à une valeur **epsilon** donné en paramètre, en utilisant la formule suivante :

$$\frac{\pi}{2} = \frac{2}{1} * \frac{2}{3} * \frac{4}{3} * \frac{4}{5} * \frac{6}{5} * \frac{6}{7} * \ldots * * \frac{2n}{2n-1} * \frac{2*n}{2n+1}$$

## Exercice N°2

On désire concevoir l'interface graphique suivante :




L'utilisateur commence par saisir une valeur d'epsilon comprise entre $10^{-6}$ et $10^{-2}$ et après avoir cliqué sur le bouton « **Approximer** », une valeur approchée de π sera calculée puis affichée dans le label correspondant en utilisant la formule suivante :

$$\pi = \frac{2}{1!} + \frac{(1!)^2 2^2}{3!} + \frac{(2!)^2 2^3}{5!} + \frac{(3!)^2 2^4}{7!} + \dots = \sum_{n=0}^{\infty} \frac{2^{n+1}(n!)^2}{(2n+1)!}$$

## Exercice N°3

La suite $(V_n)$ définie sur N* par $V_n = \frac{\text{Fib}(n+1)}{\text{Fib}(n)}$ semble converger vers $\lambda = \frac{1+\sqrt{5}}{2}$ appelé nombre d'or, dont une valeur approchée est 1,618.

Soient deux suites U et V définies à partir de :

$U_1 = 1$ et $U_2 = 2$

$U_i = U_{i-1} + U_{i-2}$ pour $U \geq 3$

$V_i = U_i / U_{i-1}$ pour tout $i \geq 2$

La suite $V_n$ tend vers une limite appelée nombre d'or.

On suppose que le $n^{ième}$ terme de la suite V soit $V_n$, donc un nombre approché du nombre d'or avec une précision e dès que $|V_n - V_{n-1}| < e$.

Ecrivez un programme Pascal nommé Nombre_Or, qui cherche $V_n$ à $10^{-4}$ près et son rang.

Good Luck