Exercice 01 :

Fonction qui calcule le volume d'un cône :

```
function [v]=volucone(h, r)
   v=%pi*h.*r.^2/3
endfunction
```

Programme qui permet d'introduire une liste de données des paramètres d'entrée de la fonction qui calcule le volume d'un cône :

```
clear,clc // Ecraser les variables déclarés, Effacer le console.
r=[1,2,3,4,5]
h=[0.5,0.6,0.7,0.8,0.9]
R=diag(r.^2)// Diagonaliser la matrice [r].
H=diag(h)// Diagonaliser la matrice [h].
exec('C:\...\...\Documents\volucone.sci', -1)// Exécuter la fonction volucone(h,r).
 / /Bien préciser le chemin de son fichier : C: \... \...\volucone.sci
vcone=volucone(H,R)//Appeler la fonction volucone(h,r).
x=vcone>0//Générer une matrice booléenne de la matrice [vcone] pour localiser les valeurs calculées par la fonction (valeurs 'T').
vcone(x)//Appliquer le filtre booléen [vcone] sur la matrice des valeurs calculées par la fonction, pour les regrouper dans un seul vecteur vcone(x).
disp(vcone(x))// Afficher les valeurs vcone(x).
```

Exercice 02 :

Fonction qui calcule le volume d'un trapèze de coté b et B de la base de hauteur h :

```
function [A]=airetrap(b, B, h)
   v=(b+B)*h/2
endfunction
```

Exercice 03 :

Programme qui permet d'introduire une série de paramètres, de la hauteur et du rayon, du cône, en utilisant la fonction qui calcule son volume :

```
clear, clc // Ecraser les variables déclarés, Effacer le console.
r=[1,2,3,4,5]
h=[0.5,0.6,0.7,0.8,0.9]
R=diag(r)// Diagonaliser la matrice [r].
H=diag(h)// Diagonaliser la matrice [h].
exec('C:\Users\win7\Documents\volucone.sci', -1)// Exécuter la fonction volucone(h,r).
vcone=volucone(H,R)//Appeler la fonction volucone(h,r).
x=vcone>0//Générer une matrice booléenne de la matrice [vcone] pour localiser les valeurs calculées par la fonction (valeurs 'T').
vcone(x)//Appliquer le filtre booléen [vcone] sur la matrice des valeurs calculées par la fonction, pour les regrouper dans un seul vecteur vcone(x).
disp(vcone(x))// Afficher les valeurs vcone(x).
```

Programme qui calcule les volumes pour un nombre définie de solide en utlisant l'instruction "Select … case » :

```
//Programme pour calculer les volumes des solides.
clear,clc
cont=%t // Pour que le programme se répète automatiqument.
while cont do,
   solide=input("Choisir un solide pour calculer son volcume!?");
   if (solide>0)&(solide<=4) then
      select solide,
         case 1 then ac=input ("coté du cube!?"),volucube = ac^3, disp(volucube),
         case 2 then
         ap=input ("Coté du parallélograme!?"),
         hb=input ("Hauteur de la base!?"),
         hp=input ("Hauteur du parallélépipède!?"),
         voluparppd=ap*hb*hp,
         disp(voluparppd),
         case 3 then
         h=input("Hauteur du cône!?"),
         r=input("Rayon de la base du cône!?"),
         exec('C:\Users\win7\Documents\volucone.sci', -1),
         vcone=volucone(h,r),
         disp(vcone)
         case 4 then rs=input("Rayon du sphère!?"),
         vsphere=4*%pi*rs^3/3,
         disp(vsphere),
      end,
   else
      revoir=%t // Pour que le programme oblige l'utilisateur à saisir une réponse bien
      //définie (Oui ou Non). Ailleurs, il répète la demande de continuer le programme
      // ou l'interrompre.
      while revoir do // C'est la même que l'expression (while revoir=%t do).
         rep=input ("Voulez-vous continuer (o pour Oui ou n pour Non)!?","string")
         if (rep=="o")|(rep=="n") then revoir=%f, end,
         if rep=="o" then cont=%t, else cont=%f,   end,
      end,
   end
end
```

Exercice 04 :

Programme qui résout une équation du 2ème degré :

```
function [xa, xb]=equa2Deg(a, b, c)
   if (a<>0)&(b<>0)&(c<>0) then
      delta=b^2-4*a*c,
      x1=(-b-sqrt(delta))/2*a,
      x2=(-b+sqrt(delta))/2*a,
      xa=x1;
      xb=x2;
```

```
            elseif (a==0)&(b<>0)&(c<>0) then xa=-c/b;xb="none";
            elseif (a==0)&(b==0)|(c==0) then xa="none";xb="none";
          end,
        endfunction
```

Exercice 08 :

Programme qui détermine si un nombre 'n' est premier ou pas, en sa forme la plus détaillée :

```
clear,clc
cont=%t
while cont do
n=input('Saisir un nombre pour savoir s"il est premier!?')
if n==1 then printf('1 n"est pas premier')
 elseif (n>1)&(modulo (n,1)==0) then
  t=2;
  rep=%f;
  while (t<sqrt(n)) & (rep==%f)
   if modulo(n,t)<>0 then
     t=t+1;
   else
      printf('%i',n);
      printf(' n"est pas premier.');
      rep=%t
   end
  end
   if (t>=sqrt(n)) & (rep==%f) then printf('%i',n);
     printf(' est premier.');
   end
  else
      revoir=%t // Pour que le programme oblige l'utilisateur à saisir une réponse bien
      //définie (Oui ou Non). Ailleurs, il répète la demande de continuer le programme
      // ou l'interrompre.
      while revoir do // C'est la même que l'expression (while revoir=%t do).
         rep=input ("Ce n"est pas un entier! Voulez-vous continuer (o pour Oui ou n pour
Non)!?","string")
          if (rep=="o")|(rep=="n") then
             revoir=%f;
          end,
          if rep=="o" then cont=%t;
          else cont=%f;
           abort;
          end,
      end,
 end,
end
```

Sous sa forme la plus simple et un peu modifié en son algorithme :

```
n = input ("Valeur de n ?");
```

```
t = 2; //premier diviseur candidat de (n) : n=t*r, implique que t<=sqrt(n) ou r<=sqrt(n)
while (modulo(n,t) <> 0) & (t <= sqrt(n))// tant que (t) ne dévise pas (n)
   //et est inférieur à la racine carrée de (n),
t = t+1;//on passe au candidat suivant (t+1)
end
if (t > sqrt(n)) & (n > 1) then // si l'on a dépassé sqrt(n) sans trouver de diviseur,
   //c'est que (n) est premier.
printf('%i',n), printf(' est premier')
else
   //Si non, c'est à dire que (n) a eu au moins un diviseur (t),
   //c'est que (n) est n'est pas premier
printf('%i',n), printf(' n"est pas premier')
end
```

A suivre…/…

Ce document est en ébauche. Ne pas rater les mises qui seront faites sur le même document, chaque fois que l'occasion se présente.

Merci de nous informer de toute erreur ou oubli.

hajomarcontact@gmail.com