

PROGRAMMER LES RACINES DE L'ÉQUATION DU SECOND DEGRÉ

- DISCIPLINE : Mathématiques
- COURS : Calcul différentiel (201-NYA-05)
- LOGICIEL UTILISÉ : Maple
- NATURE : Programmation
- THÈME PRINCIPAL ABORDÉ : Les solutions de l'équation du second degré
- OBJECTIF(S) PÉDAGOGIQUE(S) : Maîtriser la démarche algorithmique.
Maîtriser la commande alternative
- DURÉE APPROXIMATIVE : 15 minutes
- AUTEUR : Philippe Etchecopar
- COURRIEL/TÉLÉPHONE : etchecop@globetrotter.qc.ca
-
- COLLÈGE OU UNIVERSITÉ D'ORIGINE : Cégep de Rimouski

ÉNONCÉ DU THÈME À PROGRAMMER

Utilisez la commande conditionnelle pour composer un programme prévoyant les éventuelles solutions de l'équation du second degré $ax^2 + bx + c = 0$ selon les valeurs des coefficients a , b et c .

GUIDE DU TUTEUR

1. Analyse

Nous savons que la solution de cette équation est donnée par

$$\text{sol} = \left\{ \frac{-b + \sqrt{b^2 - 4ac}}{2a}, \frac{-b - \sqrt{b^2 - 4ac}}{2a} \right\}, \text{ si } \Delta = b^2 - 4ac \geq 0.$$

Les paramètres doivent être les coefficients a , b et c . Les variables que nous devrions obtenir devraient être les solutions, si elles existent.

Comme les solutions dépendent de trois possibilités, $\Delta < 0$, $\Delta = 0$ ou $\Delta > 0$, nous devons utiliser la version de l'alternative permettant trois possibilités. De plus nous devons au préalable vérifier que $a \neq 0$.

2. Schéma (organigramme)

Dans le cas où il existe plus de deux possibilités de choix, nous devons utiliser la version ci-dessous de la structure alternative :

```
if condition_1 then  
séquence_1;  
elif condition_2 then  
séquence_2 ;  
elif condition_3 then  
séquence_3 ;  
elif condition_4 then  
séquence_4 ;  
else  
séquence_5;  
fi;
```

Cela signifie que si la *condition_1* est vérifiée, *Maple* exécutera la *séquence_1*, si elle n'est pas vérifiée et si la *condition_2* est vérifiée, elle exécutera la *séquence_2* et ainsi de suite. Il faut remarquer que pour mieux « visualiser » chacune des conditions et la séquence des instructions qu'elle implique, nous avons décalé chaque séquence d'instructions par rapport à sa condition.

Représentons ci-dessous le schéma de cette structure alternative. Nous devons d'abord déterminer les alternatives pour les cas où $a = 0$ ou $a \neq 0$. Si $a \neq 0$, le choix doit porter sur les trois possibilités concernant Δ . Cela se traduit sur le schéma ci-dessous :

Structure	Pseudo-code
<pre> graph TD Start([Alternative]) --> D1{a = 0} D1 -- V --> E1[Équation degré 1] D1 -- F --> D2{a ≠ 0, Δ < 0} D2 -- V --> E2[Pas de solutions] D2 -- F --> D3{a ≠ 0, Δ > 0} D3 -- F --> S1["sol = [-b / (2a)]"] D3 -- V --> S2["sol = [(-b ± √Δ) / (2a)]"] S1 --> End([]) S2 --> End </pre>	<p><i>Initialisation</i></p> <p><i>Condition</i> Si $a = 0$ <i>Séquence vraie</i> Équation degré 1 <i>Séquence fausse</i> <i>Condition</i> Si $a \neq 0, \Delta < 0$ <i>Séquence vraie</i> Pas de solutions <i>Séquence fausse</i> <i>Condition</i> Si $a \neq 0, \Delta > 0$ <i>Séquence vraie</i> Deux solutions <i>Séquence fausse</i> Solution double</p>

3. Traduction en langage Maple et validation

Ce schéma se traduit en code *Maple*. Pour valider le programme, nous calculerons, si elles existent, les racines de différentes équations.

```
SD:=proc(a,b,c)
local delta, x1, x2, solutions;
delta:=b^2-4*a*c;
if a=0 then
  print(`Ce n'est pas une équation du second degré`);
fi;

if a<>0 and delta<0 then
  solutions:={};
elif a<>0 and delta=0 then
  x1:=-b/(2*a);
  solutions:={x1};
elif a<>0 and delta>0 then
  x1:=(-b+sqrt(delta))/(2*a);
  x2:=(-b-sqrt(delta))/(2*a);
  solutions:={x1, x2};
fi;
if a<>0 then
  RETURN(solutions);
fi;
end;
```

La validation nous donne :

<code>SD(0, 1, 2); SD(1, 0, -4); SD(1, -5,</code>	<i>Ce n'est pas une équation du second deg</i> <i>{-2, 2}</i> <i>{2, 3}</i>
---	---

4. Évaluation

Il y a d'autres possibilités de programmation, en utilisant une suite de commandes alternatives à deux choix. Il faudrait évaluer les programmes les plus simples et qui prennent le moins de temps.

```
restart;
"Variables données";
f:=piecewise(x<-2, x, x<2, x^2-6, x);
a:=2;
"Graphique";
plot(f, x=-5..5, discont=true, color=red);
"Limites";
limit(f, x=2);
```

Objectifs (notions et concepts visés)

Primaires : *Maîtrise de la démarche algorithmique*

Secondaires : *Maîtrise de la commande alternative*

Synthèse et question(s) de relance

Évaluer les extremums d'une fonction.