Dichotomie.

I Boucle « Tant que ».

Une *boucle* en programmation est un groupe d'instructions qui est répété plusieurs fois.

Commande « Tant que ».

La commande « Tant que » permet de répéter une commande tant qu'une certaine condition est vérifiée.

Par exemple la phrase « Je ferai des exercices de mathématique tant que je n'aurais pas la moyenne » signifie que je recommencerai la commande « faire des exercices de mathématique » aussi longtemps que la condition « je n'ai pas la moyenne » est vraie.

Exercice 1

Donnez le tableau d'état des variables de l'algorithme suivant :

```
1 S prend la valeur 0
2 tant que S < 10 faire
3 |  S prend la valeur S + 3
4 fin
5 afficher S
```

Avec l'exemple du programme de l'exercice précédent nous retiendrons que le tant que nécessite une condition (un test d'arrêt) et une commande, ou une liste de commande, qui peut être répétée.

♡

Langage naturel	Langage Ti	Python
1 **tant que** $S < 10$ **faire** 2 | S **prend la valeur** $S + 3$ 3 **fin**	:While $S < 10$:$S + 3 \to S$:End	**while** S<10: s=s+3

Utilisation d'un compteur.

Les *compteurs* en programmation sont des variables du programme qui servent, par exemple, à compter le nombre de fois qu'une boucle a été réalisée.

Exercice 2

Étant donné une entier naturel non nul $n \in \mathbb{N}$ nous définirons *factoriel n* comme l'entier naturel renvoyé par la fonction de Python ci-contre.

```
def factorielle(n):
    k=1
    produit=1
    while k<n:
        produit=k*produit
        k=k+1
    return(produit)
```

1. Donnez le tableau d'état des variables si je calcul *factorielle*(3).
2. Expliquez en quoi k peut être appelé un compteur.
3. Retranscrivez ce programme sur votre calculatrice.

II Un jeu.

Règles du jeu.

On demande à l'utilisateur de deviner en moins de six essais un nombre tiré au hasard entre 10 et 100.

On lui indique à chaque fois si le nombre qu'il a proposé est supérieur, inférieur ou égale au nombre cherché.

Programmation du jeu.

Exercice 3

1. Programmez le jeu précédent en langage naturel puis sur la Ti82 : l'utilisateur du programme essayant de deviner le nombre choisi par la calculatrice.
   Vous pourrez utiliser la fonction aléatoire *Entalea(borne inférieure, borne supérieure)* qui renvoie un entier choisi aléatoirement entre les entiers *borne supérieure* et *borne inférieure*.
2. Comment modifier le programme pour rendre le jeu plus simple ou plus compliqué ?
3. Modifiez le programme afin qu'il indique le nombre d'essais autorisés pour trouver la réponse.
4. Quelle stratégie faut-il développer pour trouver la réponse ?

De façon générale lorsqu'un raisonnement ou un algorithme sépare un ensemble en deux parties nous parlerons d'un procédé de *dichotomie*.

III Dichotomie.

L'*algorithme de bissection* est un algorithme de dichotomie qui permet de trouver une valeur approchée du zéro d'une fonction (valeur de x qui annule la fonction) lorsque la fonction est continue (c'est-à-dire que son graphe est d'un seul tenant, d'un seul trait).

Voici un exemple de recherche d'un tel zéro : en ligne ou fichier geogebra.

Exercice 4

On considère l'algorithme ci-dessous.

```
def dichotomie(a,b,ecart):
        while b-a>ecart:
                milieu=(a+b)/2
                if milieu**2-7<0:
                        a=milieu
                else:
                        b=milieu
        return(a,b)
```

1. Programmez l'algorithme sur la Ti82.
2. (a) Lancer le programme et notez les résultats obtenus en remplaçant 0,01 par 0,001, 0,0001, 0,00001.
   (b) Donnez une valeur approchée avec la calculatrice de $\sqrt{7}$. Que remarquez-vous ? Quel semble être l'objet de cet algorithme ?
3. Justifiez le choix du nom « milieu » pour la variable qui apparaît dans l'algorithme.

IV La boucle « Pour ».

V Exercices.

Les exercices qui suivent mettent en œuvre des boucles « tant que » et pas uniquement des dichotomie.

Exercice 5

La fonction *partie entière*, notée « ent() », associe à tout nombre x positif, un nombre N calculé par l'algorithme ci-contre.

1. Calculez le résultat pour x pris dans $\{2;\ 3{,}1;\ 9{,}8\}$
2. Définissez par une phrase la fonction $x \mapsto ent(x)$, définie pour $x \in [0;+\infty[$.

```
Entrée
    Saisir x
    Affecter 0 à N
Traitement
    Tant que N + 1 ≤ x faire
        Affecter N + 1 à N
    Fin Tant que
Sortie
    Afficher N
```

Remarque. La fonction $ent()$ est déjà programmée sur la Ti82 : math, (NUM), 5 :ent(

Exercice 6

On donne le tableau de variation d'une fonction f définie sur l'intervalle $[-10;10]$. On admet que la fonction f est continue, *i.e.* que le graphe de f peut être dessiné d'un seul trait.

x	-10		-5		3		10
$f(x)$	-260	↗	165	↘	-91	↗	840

1. Combien l'équation $f(x) = 0$ a-t-elle de solutions ?
2. Déterminez, suivant la valeur de $k \in \mathbb{R}$, le nombre de solution de l'équation $f(x) = k$.
3. L'équation $f(x) = 0$ admet une solution dans l'intervalle $[-5;3]$.
   Lequel de ces deux algorithmes permet de trouver une valeur approchée de cette solution ? Justifiez.

```
Entrée
    Affecter −5 à A
    Affecter 3 à B

Traitement
    Tant que B − A > 0,01 faire
        Si f((A+B)/2) < 0 alors
            Affecter (A+B)/2 à A
        Sinon
            Affecter (A+B)/2 à B
        FinSi
    Fin Tant que

Sortie
    Afficher (A+B)/2
```

```
Entrée
    Affecter −5 à A
    Affecter 3 à B

Traitement
    Tant que B − A > 0,01 faire
        Si f((A+B)/2) < 0 alors
            Affecter (A+B)/2 à B
        Sinon
            Affecter (A+B)/2 à A
        FinSi
    Fin Tant que

Sortie
    Afficher (A+B)/2
```

Exercice 7 pour s'entraîner.

L'*algorithme d'Euclide* permet de calculer le PGCD de 391 et 221 (le plus grand entier naturel qui divise à la fois les deux entiers).

$$391 = 221 \times 1 + 170$$
$$221 = 170 \times 1 + 51$$
$$170 = 51 \times 3 + 17$$
$$51 = 17 \times 3 + 0$$

Le PGCD de 391 et 221 est 17 (le dernier reste non nul). La première égalité $391 = 221 \times 1 + 170$ traduit la division euclidienne de 391 par 221. D'une manière générale la division euclidienne de a par b se traduit par l'égalité $a = b \times q + r$.

Cet algorithme se programme de la façon ci-contre en Python.
$a\%b$ désigne le reste de la division euclidienne.
$!=$ signifie différent ($\neq$).

Il n'y a pas de commande spécifique sur la Ti 82 pour calculer le reste de la division euclidienne de a par b. Vous pourrez utiliser la fonction $ent()$ de la calculatrice.

Programmez cet algorithme sur la Ti 83.

```
def pgcd(a,b):
        while a%b != 0:
                reste= a%b
                a=b
                b=reste
        return(b)
```

Exercice 8 pour s'entraîner.

Pierre place la somme de 5 000 € sur un compte épargne rémunéré à 2 % par an. Chaque année, les intérêts s'ajoutent à son capital. Il compte aussi placer 200 € de plus par an. Il souhaite savoir au bout de combien d'années son épargne dépassera 10 000 €.

1. Complétez le tableau.

Année 0	Année 1	Année 2	Année 3
5 000 €	5 300 €		

2. On note S la fonction qui à n, entier naturel, associe le montant de l'épargne au bout de n années ($n \neq 0$).
   Chloé conjecture que l'expression de S est
   $$S(n) = n \times 5000 \times 1{,}02 + n \times 200$$
   Est-ce la bonne expression de S ? Justifiez votre réponse.
3. Pierre décide de trouver la solution à l'aide d'un algorithme.
   Aidez le en complétant l'algorithme suivant.

```
def attente(objectif)
        capital=5000
        annee=1
        while capital<=objectif:
                capital=...
                annee=annee+1
        return(annee)
```

Exercice 9 pour s'entraîner.

Un prince indien très riche demanda à ses sages de lui inventer un divertissement. Quelque temps plus tard, le brahmane Sissa lui apporta un nouveau jeu : les échecs.

Ce jeu passionna le prince, il y joua des journées entières. Pour remercier Sissa, il lui promit une récompense de son choix.

Le brahmane demanda la quantité de grains de blé nécessaire pour remplir l'échiquier de la façon suivante :

— on place un grain sur la première case,
— deux grains sur la seconde,
— quatre sur la troisième,
— huit sur la quatrième,
— ...

en doublant le nombre de grains jusqu'à la soixante-quatrième case de l'échiquier.

Le prince trouva cette demande bien modeste. Est-ce le cas ?