

Sujets de travaux pratiques

<http://revue.sesamath.net/spip.php?article902>

patrick.raffinat@univ-pau.fr

Les séances durent une heure 30, mais les sujets sont volontairement trop longs pour la plupart des élèves ... qui doivent donc terminer les Tps d'ici la séance suivante !

Remarque :

La longueur des Tps est à relativiser car de nombreux exercices ont été préalablement traités (en notation algorithmique) en cours ou en travaux dirigés.

Table des matières

TP 1 : PROGRAMMATION VISUELLE ET TORTUE.....	3
TP 2 : PROGRAMMATION VISUELLE ET INSTRUCTIONS CONDITIONNELLES.....	5
TP 3 : INTRODUCTION À JAVASCOOL.....	7
TP 4 : BOUCLES EN JAVASCOOL.....	9
TP 5 : SOUS-PROGRAMMES AVEC LA TORTUE.....	11
TP 6 : SOUS-PROGRAMMES EN JAVASCOOL.....	13

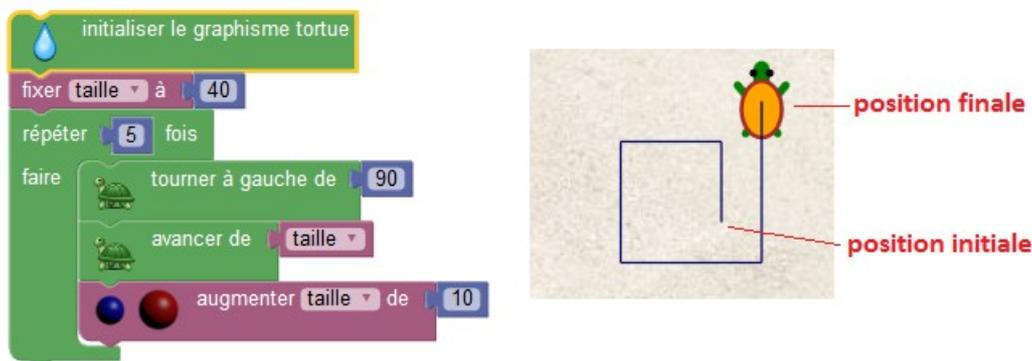
TP 1 : programmation visuelle et tortue

Pour faire de la programmation visuelle, vous utiliserez la suite logicielle Mathem@ALGO, disponible à l'adresse <http://raffinat.perso.univ-pau.fr/mathemaALGO/index.html>. Pour y écrire des programmes avec la tortue, vous cliquerez ensuite sur le lien « tortue » dans l'onglet « Blockly/Xcas ».

Remarque : le TP est à terminer pour la séance suivante...

Exercice 1 : prise en main

1) Téléchargez (bouton Démo1) le premier exemple disponible, puis exécutez-le après avoir changé le nombre de répétitions (5 au lieu de 20)



2) Echangez (avec des clics-glissés) les instructions « tourner à gauche » et « avancer ».

3) Supprimez l'instruction « tourner à gauche » (en la déplaçant dans la poubelle ou dans le menu), puis remplacez-la par « tourner à droite » (menu « tortue », sous-menu « rotations »).

4) Supprimez l'instruction « augmenter », puis remplacez-la par :



Indications :

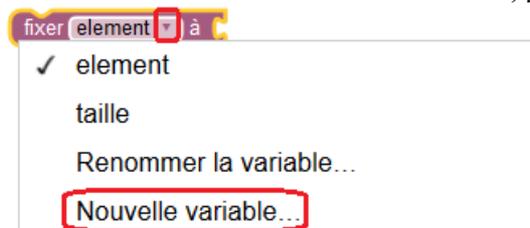
- le bloc « fixer » est dans le menu « variables », une variante possible étant de dupliquer l'instruction « fixer taille à 40 » (clic droit, puis dupliquer) et de modifier la copie.
- le bloc « + » (ou « - », « * », « : ») est dans le menu « math », sous-menu « opérations »
- le nombre « 10 » peut-être obtenu à partir du menu « math », sous-menu « constantes »

5) Faites lire la taille (menu « entrées-sorties »), au lieu de la fixer à 40



6) Faites lire l'incrément dans une variable nommée « delta », au lieu de le fixer à 10.

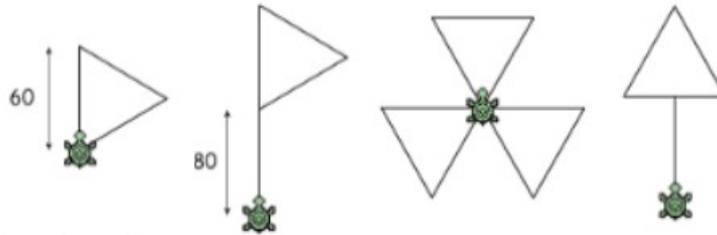
Indication : choisir le bloc « fixer element » dans le menu « variables », puis



7) Collez le programme obtenu dans le compte-rendu.

Exercice 2 : triangles

Ecrivez un programme pour réaliser chacune des figures suivantes :

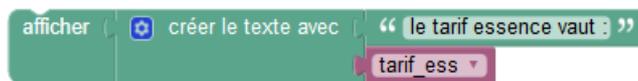


Exercice 3 : location de voitures

Un organisme de location de voitures propose à ses clients deux tarifs :

- tarif essence : 15 euros par jour de location et 85 centimes par kilomètre.
- tarif diesel : 16 euros par jour de location et 66 centimes par kilomètre.

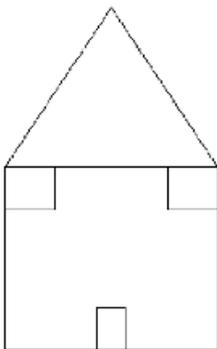
Ecrivez un programme calculant les deux tarifs. Pour l'affichage, il faudra utiliser le bloc « créer le texte » (menu « Texte ») afin de réunir une variable et un texte :



Ce programme devra être sauvegardé (bouton « Sauver »), car il sera à modifier dans le TP2. Comme Mathem@ALGO est un logiciel fonctionnant avec un navigateur web, cela entraîne quelques complications pour des raisons de sécurité :

- le programme est sauvé dans le dossier « Téléchargements », seul dossier autorisé par le navigateur.
- il faut donc ensuite transférer le code sauvé dans votre espace de stockage « S »

Exercice 4 : maison avec carrés, triangles et rectangles



La maison est formée :

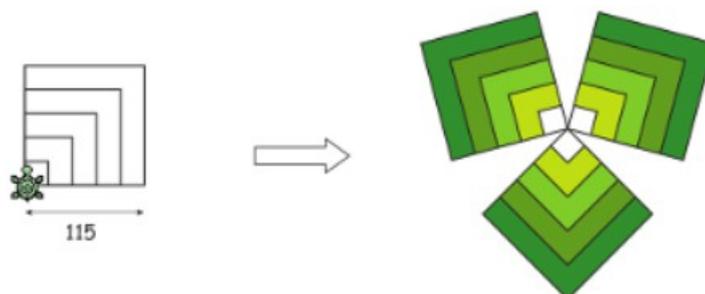
- de 3 carrés (dont les 2 fenêtres)
- d'un rectangle (la porte)
- d'un triangle équilatéral (le toit)

Remarque 1 : vous introduirez des variables, afin de permettre d'adapter facilement le dessin

Remarque 2 : comme votre programme ne sera guère lisible, vous complétez avec le logiciel Paint la copie d'écran du programme en indiquant les instructions permettant de dessiner le toit, les instructions permettant de dessiner la porte...

Exercice 5 : carrés

Réalisez les figures suivantes (sans le remplissage en vert pour la figure de droite) :



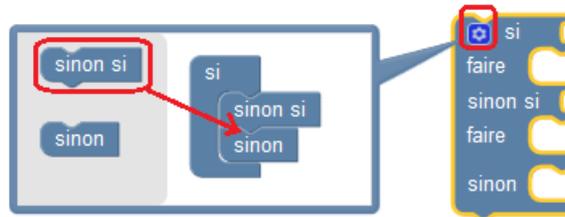
TP 2 : programmation visuelle et instructions conditionnelles

Pour faire de la programmation visuelle, vous utiliserez la suite logicielle Mathem@ALGO, disponible à l'adresse <http://raffinat.perso.univ-pau.fr/mathemaALGO/index.html>.

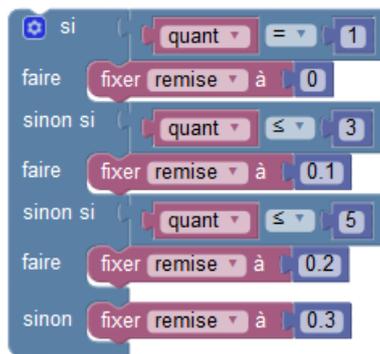
Exercice 1 : prise en main

Dans le menu « conditionnelles », vous trouverez des blocs pour un « si » simple, un « si-sinon » et un « si-sinon-si-sinon ». On peut leur ajouter des nouvelles branches avec la manipulation suivante :

On fait glisser le sinon si entouré

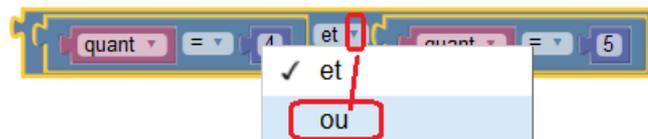


1) Construisez le bloc ci-dessous, puis complétez-le afin qu'il lise la quantité et affiche la remise :



Remarque : vous pouvez gagner du temps en dupliquant le bloc « quant=1 » afin d'obtenir les blocs « quant<=3 » et « quant<=5 ».

2) Pour vous habituer à manipuler des conditions plus complexes, remplacez « quant<=5 » par « quant=4 ou quant=5 » :

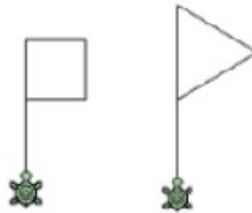


Exercice 2 : location de voitures

Récupérez le programme de l'exercice 3 du TP1, puis ouvrez-le (bouton Ouvrir) et complétez-le avec une instruction conditionnelle indiquant s'il est plus intéressant pour le client de choisir une voiture essence ou une voiture diesel.

Exercice 3 : triangle ou carré

Ecrivez un programme dessinant soit un fanion carré, soit un fanion triangulaire, en fonction de la réponse de l'utilisateur. Si celle-ci n'est pas « carre » ou « triangle », un message d'erreur doit s'afficher.



Exercice 4 : indice de masse corporelle

Ecrivez un programme qui, pour une taille exprimée en mètres et un poids exprimé en kilogrammes, calcule l'indice de masse corporelle (poids/taille²) et donne son interprétation.

Extrait de https://fr.wikipedia.org/wiki/Indice_de_masse_corporelle

Une personne pesant **61 kg** et mesurant **1,57 m** a un IMC de $\frac{61 \text{ kg}}{1,57 \text{ m} \times 1,57 \text{ m}} \approx 24,7 \text{ kg} \cdot \text{m}^{-2}$.

Cette personne présente donc une **corpulence normale**.

Interprétation de l'IMC

IMC (kg·m ⁻²)	Interprétation
moins de 16,5	dénutrition ou famine
16,5 à 18,5	maigre
18,5 à 25	corpulence normale
25 à 30	surpoids
30 à 35	obésité modérée
35 à 40	obésité sévère
plus de 40	obésité morbide ou massive

Il faut néanmoins faire attention car cet indice ne prend pas en compte la proportion de masse musculaire ni de masse osseuse

Exercice 5 : proviseur

Un proviseur souhaite automatiser la détermination de l'avis qu'il donne pour la poursuite d'études.

Ecrivez un programme, sachant que l'avis dépend de la moyenne générale qui est la moyenne pondérée de la note de mathématiques (coefficient 2), de français (coefficient 1) et d'anglais (coefficient 1) :

- moyenne générale strictement inférieure à 11 : avis réservé.
- moyenne générale comprise entre 11 et 13 : avis favorable.
- moyenne générale strictement supérieure à 13 : avis très favorable, sauf si l'étudiant n'a pas la moyenne dans une des matières (l'avis n'est alors que favorable).

Exemple : un étudiant ayant 20 en mathématiques, 20 en français et 8 en anglais n'aura qu'un avis favorable à cause de sa note d'anglais, bien que sa moyenne soit de 17.

TP 3 : introduction à Javascool

Exercice 1 : premiers programmes en Javascool...

Javascool est un logiciel pédagogique utilisé dans les lycées, principalement en terminale S dans l'option ISN. On peut le considérer comme étant une version simplifiée du langage Java, un langage très utilisé par les programmeurs du monde entier.

- 1) Lancez le logiciel (icône Javascool sur votre bureau), puis cliquez sur l'activité **abcdAlgo**.
- 2) Recopiez dans l'éditeur le programme suivant, puis enregistrez-le, puis compilez-le (bouton **Compiler**) :

```
void main(){
    double x;
    double y;
    x = 10;
    y = 2;
    print("la somme des 2 nombres vaut : ");
    print(x+y);
    println(); // retour à la ligne pour l'affichage suivant
    print("le produit des 2 nombres vaut : ");
    print(x*y);
    println();
}
```

- 2) Une erreur est indiquée (un point virgule manque à la fin d'une ligne). Corrigez-la, puis enregistrez et compilez le programme jusqu'à ce que Javascool indique « compilation réussie », ce qui signifie :

- qu'il n'y a plus d'erreur de syntaxe.
- que le programme a été transformé dans un langage « proche de celui de votre ordinateur » (donc illisible pour vous).
- que vous pouvez exécuter le programme autant de fois que vous le souhaitez en cliquant sur le bouton **Executer**.

Exécutez le programme.

- 3) Changez les valeurs de x et de y dans le programme précédent, puis testez le nouveau programme.

Attention: avant de cliquer sur le bouton « Exécuter », vous devez veiller à compiler le nouveau programme car sinon c'est l'ancien qui s'exécutera.

- 4) Modifiez le programme en faisant saisir les valeurs de x et de y dans des boîtes de dialogue :

```
void main(){
    double x;
    double y;
    x = readDouble("saisir un premier nombre : ") ;
    y = readDouble("saisir un second nombre : ") ;
    ...
}
```

Exécutez le nouveau programme.

- 5) Dans un nouvel onglet de l'éditeur (bouton **Nouveau Fichier**), écrivez un programme calculant la moyenne de 3 nombres à saisir, puis exécutez le programme.

Exercice 2 : papetier

Un papetier vend des feuilles de format A4 par cartons. Pour une quantité commandée ne dépassant pas 20 cartons, chaque carton est facturé 10 euros. Au delà, du 21^{ième} au 50^{ième} carton, le prix unitaire des cartons **supplémentaires** passe à 9 euros et, à partir du 51^{ième} carton, à 8 euros.

Ecrire un programme Javascool calculant le prix total.

Remarque : il existe un outil qui améliore la mise en forme du programme (bouton Reformater)

Exercice 3 : gagner du temps avec Mathém@ALGO...

Voir <http://raffinat.perso.univ-pau.fr/mathemaALGO/index.html>

1) Sélectionnez l'onglet PluriAlgo et le sous-onglet Principal, afin de gagner du temps pour les entrées sorties et les déclarations de variables :

The screenshot shows the PluriAlgo interface. On the left, there is a form with the following fields:

- langage: Javascool (dropdown menu)
- entrees: km duree
- sorties: tarifEss tarifDie
- Entiers: (checkbox)
- Reels: (checkbox)
- duree: km tarifEss tarifDie
- Nouveau: (button, highlighted with a red box and an arrow pointing to the code editor)

On the right, the code editor shows the following code:

```
void main() {
    double km;
    int duree;
    double tarifEss;
    double tarifDie;
    km = readDouble( "km : " );
    duree = readInt( "duree : " );
    println( tarifEss );
    println( tarifDie );
}
```

Below the code editor, there is a text description of the problem:

Un organisme de location de voitures propose à ses clients deux tarifs :

- tarif essence : 15 euros par jour de location et 85 centimes par kilomètre.
- tarif diesel : 16 euros par jour de location et 66 centimes par kilomètre.

Recopiez le résultat obtenu (en cliquant sur le bouton Nouveau) dans l'environnement Javascool, puis complétez et exécutez le programme qui doit afficher les tarifs essence et diesel.

2) Dans un concours de saut à skis, chaque skieur est noté par 5 juges. Sa note finale est obtenue en sommant les 5 notes, puis en soustrayant à cette somme la note minimale et la note maximale.

Exemple : si les notes sont 2, 1, 1, 5, 6 alors la note finale est $(2+1+1+5+6)-1-6$, soit 8

a) déclarez dans l'onglet Principal les diverses variables du morceau d'algorithme ci-dessous, en précisant lesquelles sont à lire (rubrique entrées) et lesquelles sont à afficher (rubrique sorties)

```
lire note1, note2, note3, note4, note5
somme ← note1 + note2 + note3 + note4 + note5
// calcul du minimum (noté mini)
...
// calcul du maximum (noté maxi)
...
// calcul de la note finale
noteFinale ← somme - mini - maxi
ecrire "la note finale est : ", noteFinale
```

b) cliquez sur le bouton Nouveau, puis récupérez le code Javascool obtenu et complétez-le.

Exercice 4 : proviseur

Traitez en Javascool l'exercice 5 du TP2.

Exercice 5 : seconde suivante

Écrire un programme Javascool qui saisit un temps exprimé en jours, heures, minutes et secondes. À ce temps, vous ajouterez une seconde et vous l'afficherez en jours, heures, minutes et secondes. Il vous est interdit de passer par une phase de conversion en secondes.

TP 4 : boucles en Javascool

Exercice 1

Ecrivez un programme Javascool qui demande à l'utilisateur un nombre de départ, et qui ensuite affiche les dix nombres suivants. Par exemple, si l'utilisateur entre le nombre 17, le programme affichera les nombres de 18 à 27.

Exercice 2 : sujets de bac

1) Résolvez en Javascool cet exercice du Bac ES-L 2014 Antilles-Guyane

Le terme u_n donne une estimation du nombre d'abonnés pour l'année 2013 + n $\begin{cases} u_0 = 20 \text{ millions.} \\ u_{n+1} = 0,92u_n + 3. \end{cases}$

compléter l'algorithme suivant afin de déterminer le nombre d'années nécessaires à partir de 2013 pour que le nombre d'abonnés dépasse 25 millions.

Variables :	N un nombre entier naturel non nul U un nombre réel
Traitement :	Affecter à U la valeur 20 Affecter à N la valeur 0 Tant que affecter à U la valeur $0,92 \times U + 3$ affecter à N la valeur $N + 1$ Fin Tant que
Sortie :	Afficher

2) Résolvez en Javascool cet exercice du Bac S 2014 Antilles-Guyane

En montagne, un randonneur a effectué des réservations dans deux types d'hébergements.
Une nuit en hébergement A coûte 24 € et une nuit en hébergement B coûte 45 €. Il se rappelle que le coût total de sa réservation est de 438€.

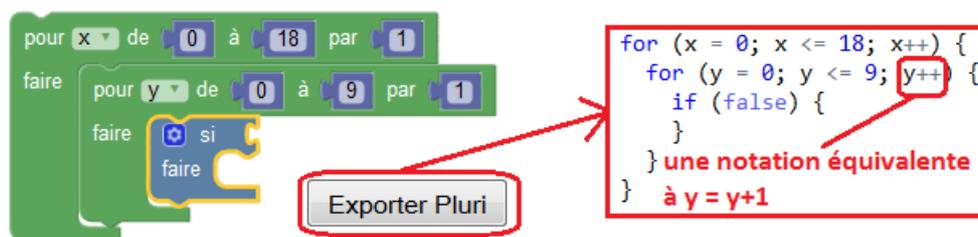
On souhaite retrouver les nombres x et y de nuitées passées respectivement en hébergement A et en hébergement B

Montrer que les nombres x et y sont respectivement inférieurs ou égaux à 18 et 9
compléter l'algorithme suivant afin qu'il affiche les couples $(x; y)$ possibles.

On l'admettra dans ce TP

Entrée :	x et y sont des nombres
Traitement :	Pour x variant de 0 ... (1) Pour y variant de 0 ... (2) Si ... (3) Afficher x et y Fin Si Fin Pour Fin Pour

Remarque : Mathém@ALGO, dont l'utilisation est facultative, aidera peut-être certains d'entre vous à écrire le code Javascool...



Exercice 3 : tables de multiplication

1) Ecrivez un programme affichant une table de multiplication par n (à saisir) sous la forme suivante :

```
voici la table de 5 :
1 fois 5 vaut 5
2 fois 5 vaut 10
...
10 fois 5 vaut 50
```

2) Ecrire un programme affichant, comme sur certains cahiers de brouillon, toutes les tables de 2 à 9

Exercice 4 : moyenne et variance

1) Ecrivez un programme qui, pour une suite de notes à saisir, calcule sa moyenne.

Remarque : le nombre de notes est aussi à saisir

2) Complétez le programme précédent pour calculer également la variance.

Indication :

Si la série statistique est de moyenne m et prend les valeurs x_1, x_2, \dots, x_n , sa variance est

$$V(X) = \frac{1}{n} \sum_{i=1}^n (x_i - m)^2 = \left(\frac{1}{n} \sum_{i=1}^n x_i^2 \right) - m^2$$

La formule en rouge ne peut pas être utilisée ici : expliquez pourquoi

Il faut donc utiliser la formule en vert

Remarque : vérifiez le résultat obtenu avec un tableur

Exercice 5 : moyenne et variance (encore !)

Reprenez l'exercice précédent en supposant que le nombre de notes n'est pas connu a priori et que l'utilisateur peut en fournir autant qu'il le souhaite. Pour signaler qu'il a terminé, on convient qu'il fournira une note fictive négative.

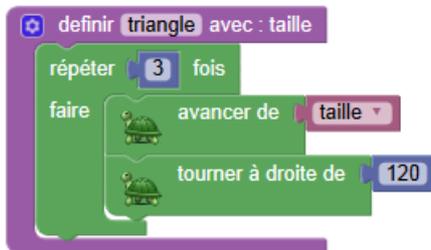
TP 5 : sous-programmes avec la tortue

Pour faire de la programmation visuelle, vous utiliserez la suite logicielle Mathem@ALGO, disponible à l'adresse <http://raffinat.perso.univ-pau.fr/mathemaALGO/index.html>.

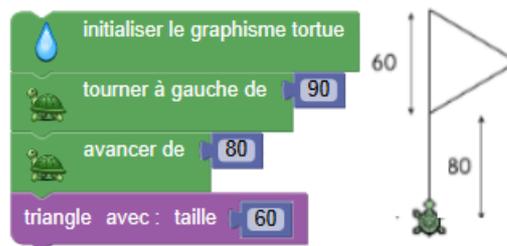
Exercice 1 : prise en main

Pour faire des figures composées de triangles (carrés...), il est conseillé d'introduire des sous-programmes (appelés procédures) :

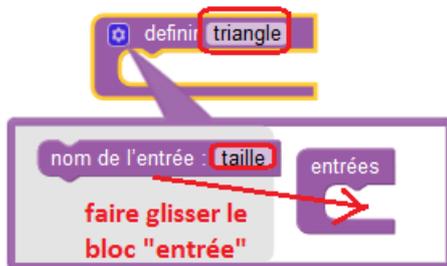
définition de la procédure



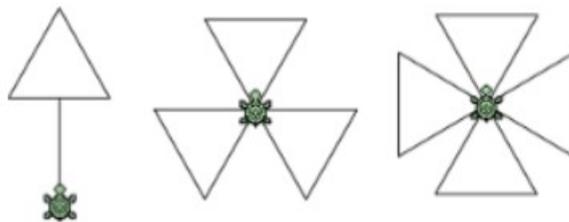
utilisation de la procédure



1) Définir la procédure Triangle



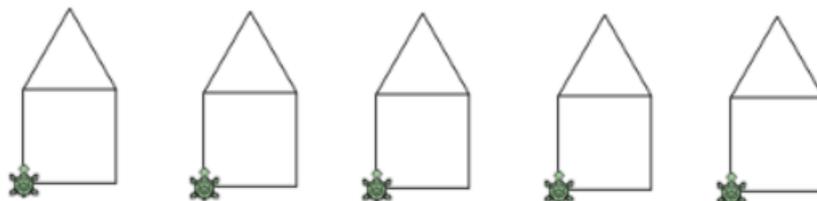
2) Utiliser la procédure Triangle pour réaliser les figures suivantes :



Exercice 2 : carrés

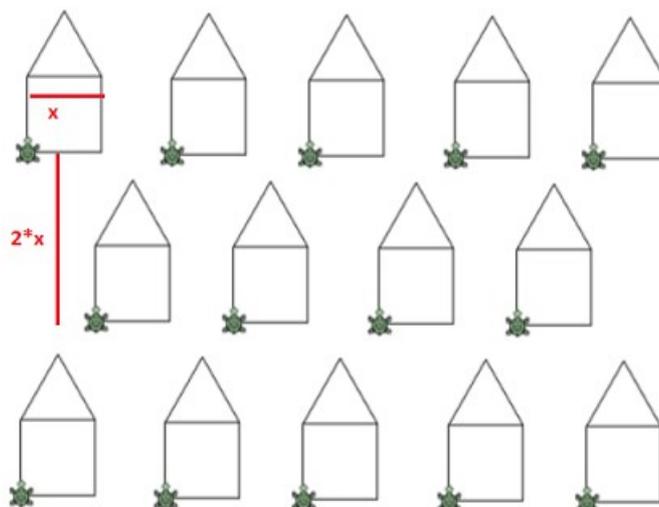
1) Définir une procédure « carré ».

2) Définir une procédure maison à partir des procédures « triangle » et « carré », puis l'utiliser afin d'afficher cinq maisons séparées par un intervalle de la taille de la maison :



Remarque : introduire une procédure nommée « rangeeDeMaisons » paramétrée par la taille d'une maison et le nombre de maisons, procédure qui sera également utilisée dans la question 3.

3) Utiliser la procédure « rangeeDeMaisons » pour construire ce lotissement avec 3 rangées de maisons :



Exercice 3 : devenez un(e) artiste !

Programmez ce magnifique dessin :

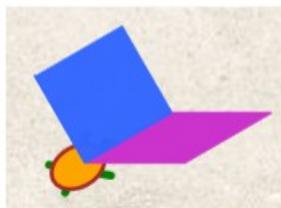


Indication : récupérez la procédure losange (bouton Démo 3 de l'extension tortue de Blockly)



definir losange avec : tail...

Inutile d'examiner le code définissant cette procédure



Remarque : il n'est pas nécessaire de respecter la couleur des carrés bleus, ce qui facilitera le travail puisqu'il n'y aura alors plus besoin de les dessiner (ils seront délimités par les losanges les bordant).

TP 6 : sous-programmes en Javacool

Exercice 1

1) Complétez la procédure afficherEntiers pour qu'elle affiche tous les entiers compris entre debut et fin :

```
void afficherEntiers(int debut, int fin) {
    // affiche tous les entiers entre les nombres debut et fin
    ...
}
```

attention : pour tester la procédure, il faut un programme principal, par exemple celui-ci

```
void main() {
    afficherEntiers(18, 27);
}
```

2) Modifiez le programme principal pour qu'il demande à l'utilisateur d'entrer la valeur de début et la valeur de fin.

3) Modifiez le programme principal pour qu'il résolve (en utilisant la procédure afficherEntiers) l'exercice 1 du TP 4, dont je rappelle l'énoncé :

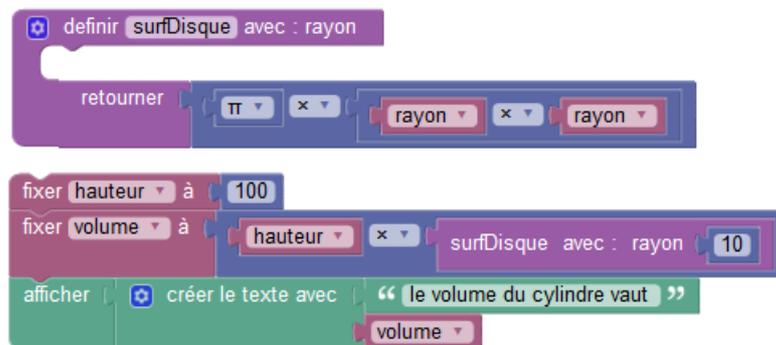
Ecrivez un programme Javacool qui demande à l'utilisateur un nombre de départ, et qui ensuite affiche les dix nombres suivants. Par exemple, si l'utilisateur entre le nombre 17, le programme affichera les nombres de 18 à 27.

Exercice 2

1) Traduisez en Javacool ce programme visuel :



2) Il est regrettable de ne pas pouvoir réutiliser la **procédure** surfDisque dans des calculs, par exemple pour calculer le volume d'un cylindre. C'est pourquoi il est préférable de définir une **fonction** surfDisque :



Traduisez en Javacool ce programme visuel.

Exercice 3 : factorielle

```

int factorielle(int n) {
    int k;
    int fact;
    fact=1;
    for(k=2; k<=n; k=k+1) {
        fact = fact * k;
    }
    return fact;
}

void main(){
    println(factorielle(3));
}

```

1) Définissez la fonction factorielle, puis testez-la avec un programme principal calculant la factorielle de 3.

2) Utilisez la fonction factorielle afin de connaître vos chances de gagner au tiercé, quarté ou quinté.

On demandera à l'utilisateur le nombre de chevaux partants (n), et le nombre de chevaux joués (p). Les deux messages affichés devront être :

- une chance sur X de gagner dans l'ordre (rappel : $X = n! / (n - p)!$)
- une chance sur Y de gagner dans le désordre (rappel : $Y = n! / (p! * (n - p)!)!$)

Exercice 4 : minimum et maximum

Voici deux fonctions calculant le minimum et le maximum de deux nombres :

<pre> double getMin(double x, double y) { if (x<y) { return x; } else { return y; } } </pre>	<pre> double getMax(double x, double y) { if (x>y) { return x; } else { return y; } } </pre>
---	---

1) Utilisez ces deux fonctions dans un programme principal calculant le minimum et le maximum de 3 nombres (à saisir).

2) Utilisez ces deux fonctions dans un programme principal calculant le minimum et le maximum de 5 nombres (à saisir).

3) Dans un concours de saut à skis, chaque skieur est noté par 5 juges. Sa note finale est obtenue en sommant les 5 notes, puis en soustrayant à cette somme la note minimale et la note maximale.

Exemple : 4 2 5 1 3 --> note finale 9 (4+2+5+1+3-1-5)

Adaptez le programme de la question 2 pour résoudre ce problème.

4) En consultant <https://docs.oracle.com/javase/7/docs/api/java/lang/Math.html>, vérifiez qu'il existe des fonctions prédéfinies pour calculer le minimum et le maximum de deux nombres :

```

double max(double a, double b)
    Returns the greater of two double values.

double min(double a, double b)
    Returns the smaller of two double values.

```

Adaptez le programme de la question 3 en abandonnant les fonctions getMin et getMax puisqu'il existe des fonctions prédéfinies pour calculer le minimum et le maximum.

5) Si on veut gérer plusieurs skieurs, mieux vaut introduire une fonction calculant la note finale :

```

double getNoteFinale(double n1, double n2, double n3, double n4, double n5) {
    ...
}

```

Complétez la fonction et écrivez un programme principal gérant une compétition de 2 skieurs.

Remarque : vous devez non seulement donner la note finale des 2 skieurs, mais aussi leur classement.