

Examen blanc

Documents autorisés

Exercice 1

Écrire les conditions opposées à celles déjà définies dans le tableau ci-dessous.

Condition	Condition inverse
(note = 20) ET (age < 25)	
(de ≠ 6) OU (nbCoup > 5)	
	((de1 = 6) OU (de2 = 6)) ET (nbCoup < 5)
	NON((de1 = 6) OU (de2 = 6))

Exercice 2

Écrire l'algorithme qui permet de calculer le montant de la prime mensuel d'un représentant.

Chaque mois les représentants perçoivent une prime (prime), composée d'une commission susceptible de faire l'objet d'une majoration, liée au montant du chiffre d'affaire du mois écoulé (camois) et à sa progression par rapport au moins antérieur (camoispr).

Les règles appliquées pour le calcul sont les suivantes :

- lorsque le chiffre d'affaires du mois est inférieur à 300 000€, il n'y a pas attribution de prime ;
- lorsque le chiffre d'affaires du mois ne dépasse pas 450 000€, seule la tranche supérieure à 300 000€ est rétribuée, le taux de commission est alors de 2 pour mille.
- Si le chiffre d'affaires est supérieur à 450 000€, le calcul précédent s'applique jusqu'à 450 000€, et la tranche dépassant les 450 000€ est commissionnée à 4 pour mille ;
- lorsqu'une prime est accordée, elle est majorée de 300€ (major) si la progression du chiffre d'affaires par rapport au mois précédent est au moins de 10%, sauf en septembre, mois pour lequel il n'y a pas de majoration ;
- dans tous les cas, la prime (commission plus majoration) est limitée à 2000€.

Exercice 3

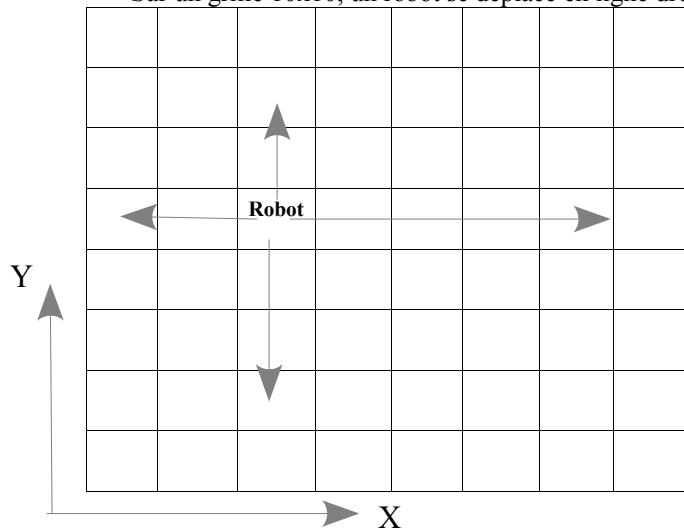
Écrire l'algorithme qui permet de calculer la durée d'un trajet connaissant l'heure de départ et d'arrivée. On se contente des heures et des minutes et la durée totale ne dépassera jamais 24 heures. Les heures et minutes seront des entiers.

Exemple : Heure de départ : 7h45

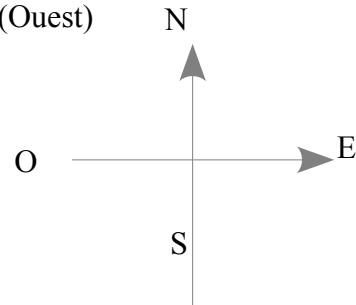
Heure d'arrivée : 21h02

Exercice 4

Sur une grille 10x10, un robot se déplace en ligne droite comme ci-dessous.



Exemples de directions : N
(Nord) ou S (Sud) ou E (Est) ou
O (Ouest)



Écrire un algorithme qui :

- demande à l'utilisateur la position initiale du robot
- puis en boucle 10 fois de suite :
 - demande la direction à prendre (N, S, E ou O) et le nombre de pas
 - puis soit calcule et affiche sa nouvelle position, soit envoie un message d'erreur si le déplacement aboutit à faire sortir le robot de la grille.
- Affiche « Gagné ! » si les 10 déplacements ont pu être effectués sans que le robot ne sorte de la grille et « Perdu après B coups ! » sinon (où B est le nombre de coups joués avant de sortir de la grille).

Remarque 1 : Évidemment si le robot aboutit hors de la grille lors d'un de ses déplacements, l'algorithme ne redemandera pas l'utilisateur de saisir un nouveau déplacement.

Remarque 2 : Si la direction de déplacement saisie est erronée, l'algorithme sortira de la boucle et renverra un message expliquant l'erreur et pourquoi l'utilisateur a perdu.

Corrige

Exo 4 :

```
Algorithme Deplacement_Robot
{
    constante entier NOMBRE_COUPS_MAX <--- 10
    variable x, y : entier // pour la position de le robot
        direction : caractere
        nombre_pas, nombre_coups : entier
        dedans, erreur : logique

    nombre_coups <--- 0

    Afficher("Saisir la position initiale du robot en x et en y : ")
    Saisir(x,y)

    Si ((x>0) ET (x<11) ET (y>0) ET (y<11)) alors
    {
        dedans <--- VRAI
        erreur <--- FAUX
    }
    sinon
    {
        dedans <--- FAUX
        erreur <--- VRAI
    }

    Tant que ((nombre_coups < NOMBRE_COUPS_MAX) ET dedans ET NON(erreur))
    {
        Afficher("Saisir la direction du déplacement à effectuer : ")
        Saisir(direction)
        Afficher("Saisir le nombre de pas dont vous voulez déplacer le robot : ")
        Saisir(nombre_pas)

        Selon (direction)
        {
            'N' : y <--- y + nombre_pas
            'S' : y <--- y - nombre_pas
            'E' : x <--- x + nombre_pas
            'O' : x <--- x - nombre_pas
            sinon : erreur <--- VRAI
        }

        Si (NON(erreur)) alors
        {
            nombre_coups <--- nombre_coups + 1
            Si ((x<1) ou (y<1) ou (x>10) ou (y>10)) alors
                dedans <--- FAUX
            sinon
                Afficher("Coup n°",nombre_coups," : le robot est maintenant en (",x," ",y,")\n")
        }
    }

    Si (erreur) alors
    {
        Si (dedans) alors
            Afficher("La direction du déplacement saisi n'existait pas. On décide donc que vous avez perdu après
les ",nombre_coups," coups précédents.\n")
        sinon
            Afficher("La position d'origine était en dehors de l'échiquier.\n")
        }
    sinon
    {
        Si(dedans) alors
            Afficher("Gagné !\n")
        sinon
            Afficher("Perdu après ",nombre_coups," coups !\n")
        }
    }
}
```