**fonction int gestionMenu()**

déclarer redirection\_menu comme entier initialisé à 0

faire

// Affiche les options du menu

afficher "Sélectionnez le numéro correspondant à vos attentes"

afficher "====================== Menu ======================"

afficher "- 0 Quitter"

afficher "- 1 Pour la conversion Binaire"

afficher "- 2 Pour la fonction trigonométrique"

// Lit la sélection de l'utilisateur

lire redirection\_menu

// Vérifie si la sélection de l'utilisateur est valide (entre 0 et 2)

si redirection\_menu est inférieur à 0 ou supérieur à 2 alors

// Si la sélection n'est pas valide, affiche un message d'erreur

afficher "Le numéro entré n'est pas correct (entre 0 et 2)"

// Répète l'affichage du menu tant que la sélection de l'utilisateur n'est pas valide

jusqu'à ce que redirection\_menu soit entre 0 et 2

// Retourne la sélection valide de l'utilisateur

retourner redirection\_menu

fin fonction

**fonction void impressionBinaire(entier valBinaire, entier optionAffichage)**

// Si optionAffichage est 0, calcule le nombre minimal de bits nécessaires

si optionAffichage est égal à 0 alors

déclarer start comme entier initialisé à 0

// Décale valBinaire vers la droite jusqu'à ce qu'il devienne 0

// Le nombre de décalages est le nombre minimal de bits nécessaires

tant que (valBinaire décalé vers la droite de start) n'est pas égal à 0 faire

incrémenter start

fin tant que

// Met à jour optionAffichage avec le nombre minimal de bits nécessaires

optionAffichage devient start

fin si

// Parcourt chaque bit de valBinaire de gauche à droite

pour i de optionAffichage - 1 à 0 décrémenter i

// Affiche le bit en position i

afficher (valBinaire décalé vers la droite de i) ET 1

// Ajoute un espace après chaque groupe de 4 bits pour une meilleure lisibilité

si i modulo 4 est égal à 0 et i n'est pas égal à 0 alors

afficher " "

fin si

fin pour

fin fonction

**fonction void impressionDecimale(flottant valBinaire)**

// Sépare la partie entière de la partie fractionnaire

déclarer int\_part comme entier initialisé à la partie entière de valBinaire

déclarer frac\_part comme flottant initialisé à valBinaire moins int\_part

// Affiche la partie entière en binaire

si int\_part est égal à 0 alors

afficher "0"

sinon

appeler impressionBinaire(int\_part, 0)

fin si

// Affiche le séparateur pour la partie fractionnaire

afficher "."

// Boucle pour convertir et afficher la partie fractionnaire

tant que frac\_part n'est pas égal à 0.0 faire

// Multiplie la partie fractionnaire par 2

frac\_part devient frac\_part multiplié par 2

// Si la partie fractionnaire est supérieure ou égale à 1

si frac\_part est supérieur ou égal à 1.0 alors

// Affiche 1

afficher "1"

// Soustrait 1 de la partie fractionnaire

frac\_part devient frac\_part moins 1.0

sinon

// Si la partie fractionnaire est inférieure à 1, affiche 0

afficher "0"

fin si

fin tant que

fin fonction

**fonction void ImpressionEtConversion(chaîne de caractères input, entier optionAffichage)**

// Convertit la chaîne de caractères en nombre flottant

déclarer num comme flottant initialisé à atof(input)

// Vérifie si le nombre est entier

si num est égal à (entier)num alors

déclarer int\_num comme entier initialisé à (entier)num

// Vérification si le nombre de bits est suffisant pour représenter l'entier

si optionAffichage n'est pas égal à 0 et optionAffichage est inférieur à (entier)(log2(abs(int\_num)) + 1) alors

afficher "Erreur : le nombre de bits spécifié est insuffisant pour représenter le nombre."

retourner

fin si

// Affiche le préfixe binaire

afficher "0b "

// Vérifie si le nombre entier est négatif

si int\_num est inférieur à 0 alors

// Convertit l'entier en non-signé pour l'affichage en binaire

déclarer bin\_num comme entier non-signé initialisé à (entier non-signé)int\_num

appeler impressionBinaire(bin\_num, optionAffichage)

sinon

// Affiche l'entier en binaire

appeler impressionBinaire(int\_num, optionAffichage)

fin si

sinon

// Si le nombre est à virgule flottante, affiche la partie entière et fractionnaire en binaire

appeler impressionDecimale(num)

fin si

// Passe à la ligne suivante après l'affichage

afficher "\n"

fin fonction

**fonction void conversionBinaire()**

déclarer input comme tableau de caractères de taille 20

déclarer option comme entier

// Demande à l'utilisateur de saisir une valeur décimale à convertir en binaire

afficher "Saisissez la valeur décimale à convertir en binaire"

lire input

// Boucle pour vérifier que l'option choisie est valide

faire

afficher "Choix de la représentation binaire"

afficher "- 1 Pour afficher sur le nombre minimum de bits"

afficher "- 2 Pour afficher sur 8 bits"

afficher "- 3 Pour afficher sur 16 bits"

afficher "- 4 Pour afficher sur 32 bits"

lire option

// Vérifie si l'option est valide (entre 1 et 4)

si option est inférieur à 1 ou supérieur à 4 alors

afficher "Le numéro entré n'est pas correct (entre 1 et 4)"

fin si

jusqu'à ce que option soit entre 1 et 4

// Utilise une structure conditionnelle pour gérer les différentes options choisies

selon option faire

cas 1:

Appeler ImpressionEtConversion(input, 0) // Appelle ImpressionEtConversion avec option 0 pour le nombre minimum de bits

cas 2:

Appeler ImpressionEtConversion(input, 8) // Appelle ImpressionEtConversion avec option 8 pour 8 bits

cas 3:

Appeler ImpressionEtConversion(input, 16) // Appelle ImpressionEtConversion avec option 16 pour 16 bits

cas 4:

Appeler ImpressionEtConversion(input, 32) // Appelle ImpressionEtConversion avec option 32 pour 32 bits

sinon

afficher "Option invalide."

fin selon

fin fonction