Convertisseur dans différentes bases numériques

I- Présentation du projet

On se propose d'écrire un programme qui nous permette de convertir les nombres binaires, décimaux, octaux, et hexadécimaux entre eux.

Nous nous appuierons pour cela sur les algorithmes connus de conversion permettant de passer d'une base à une autre.

Instructions :

- Le code devra être indenté
- Chacune des fonctions devra être commentée
- Il sera possible de consulter l'historique du développement sur ***GitLab***

II- Travail à effectuer

But : On doit pouvoir entrer un nombre en n'importe quelle base soit 2, 8, 10, 16, pour le convertir grâce a notre en programme en base 2, 8, 10, ou 16.

Aide : Il est proposé d'écrire les fonctions suivantes :

void inverseChiffres(char ch[],int nb) : cette fonction, reçoit en argument une chaine ch de taille. Cet argument est passé par référence. Elle permet d'inverser la position des chiffres dans les cases du tableau. Elle ne renvoie rien de ce fait.

int longueurChaine(char ch[]) : calcul la longueur d'une chaîne de caractères. On passe cette chaîne ne argument.

void conversionDecimalBinaire(int nombre, int nb ,char bin[]) : va convertir un nombre décimal passé en argument nb, en un nombre binaire qui est une chaîne de caractère. On récupère la conversion dans bin, passé en argument (donc par référence). La fonction ne renvoie rien.

void conversionHexadecimalBinaire(char hex[], char bin[]) : va convertir une chaîne de caractères sous forme hexadécimale passée en argument en une chaîne binaire.

void conversionOctalBinaire (char oct[], int taille, char bin[]): va convertir une chaîne de caractères sous forme octale passée en argument en une chaîne binaire.

int conversionBinaireDecimal(char bin[]) : va convertir une chaîne de caractères qui représente le nombre binaire en un entier qui correspond au nombre décimal obtenu. Ce nombre décimal est retourné par la fonction.

int conversionOctalDecimal(char oct[]) : va convertir une chaîne de caractère qui représente le nombre octal en un entier qui correspond au nombre décimal obtenu. Ce nombre décimal est retourné par la fonction.

int conversionHexadecimalDecimal(char str[]) : va convertir une chaîne de caractères qui représente le nombre hexadécimal en un entier qui correspond au nombre décimal obtenu. Ce nombre décimal est retourné par la fonction.

void conversionDecimalHexadecimal(int nombre, char hex[]) : cette fonction, reçoit en argument une chaîne de caractères dans laquelle seront placés les caractères obtenus à la suite de la conversion d'un nombre décimal (lui aussi en argument) en hexadécimal. Le tableau est passé par référence. Cette fonction ne renvoie rien de ce fait. Par contre elle rempli hex avec le résultat de la conversion.

void conversionOctalHexadecimal(char oct[],char hex[]): cette fonction, reçoit en argument une chaîne de caractères correspondant au nombre octal. On obtient la chaîne hexadécimale correspondante dans hex.

void conversionBinaireHexadecimal (char bin[],char hex[]): reçoit en argument une chaîne de caractères correspondant au nombre binaire. Elle donne la chaine hexadécimale dans l'argument hex.

void conversionBinaireOctal (char bin[], char oct[]): reçoit en argument une chaîne de caractères correspondant au nombre binaire. L'argument oct reçoit la chaîne octale. La taille de la chaine binaire est aussi passée en argument.

void conversionHexadecimalOctal (char hex[],char oct[]) : reçoit en argument une chaîne de caractères correspondant au nombre hexadécimal. L'argument oct sera rempli par la chaîne octale.

void conversionDecimalOctal(int nombre, char oct[]) : reçoit en argument une entier correspondant au nombre décimal. On passe un tableau de caractères en argument pour y placer les caractères octaux, résultat de la conversion.

void correspondanceBinaireOctal(char bin[],char oct[]) : reçoit en argument une chaîne binaire. Elle remplit une chaîne de caractère octale oct.

char chiffreDecimalVersOctal(int n) :fait correspondre à un chiffre décimal le caractère octal correspondant et le renvoie.

int octalVersChiffreDecimal(char c) : fait correspondre à un chiffre octal c représenté par un caractère le chiffre décimal correspondant et le renvoie.

int hexadecimalVersChiffreDecimal(char c) : fait correspondre à un hexadécimal représenté par un caractère le chiffre décimal correspondant et le renvoie.

char chiffreDecimalVersHexadecimal(int n) : fait correspondre à un décimal représenté par un entier le chiffre Hexadécimal correspondant et le renvoie.

void hexadecimalVersChiffreBinaire(char c, char bin[]) : fait correspondre à un hexadécimal représenté par un caractère le nombre binaire correspondant et donne la chaîne binaire correspondate dans bin.

void OctalVersChiffreBinaire(char c,char bin[]): fait correspondre à un chiffre octal représenté par un caractère le chiffre décimal correspondant.

De plus, il est demandé d'effectuer un menu qui permette à un utilisateur de votre programme de choisir le type opération qu'il souhaite effectuer. On proposera ainsi différentes options à l'utilisateur, qui lui permettront d'obtenir son résultat.

Evaluation :

Historique du développement

Gestions de versions
des jalons marquent le développement - les recherches et fonctionnalités sont développées sur des branches - un état cohérent est maintenu sur une branche maîtresse - les commits sont explicites

Fonctionnement du programme

Exécution et bon fonctionnement
l'application fonctionne sans s'interrompre - pas de bogue génant l'utilisation - pas de manipulation complexe pour exécuter l'application

Traitement des erreurs
l'utilisateur est-il prévenu et/ou guidé lors d'une erreur - levée d'exceptions, gestion d'arrêt et/ou reprise du programme

Code source

Compréhension du langage / environnement
syntaxe - concepts - bonne utilisation du paradigme

Ergonomie
les conventions du langage utilisé sont appliquées (nommage, motifs) ? - utilisabilité et réutilisabilité du code (factorisation...)

Lisibilité
clarté du formatage du code (espacements, indentation, retours de ligne...) - pas de code inutilisé gênant la lecture

Documentation

présence d'une documentation - précision de la documentation sur les méthodes implémentées (comment les utiliser ou ne pas les utiliser, leur signature...)

Organisation des sources
présence d'une arborescence, nomenclature, espaces de noms, packages etc.

Documentation

Présentation du contexte
pour qui, pour quoi, comment, quand, où ?

Notice d'installation
description claire de l'environnement logiciel / matériel et des étapes à effectuer pour installer et exécuter l'application

Documentation utilisateur
permet la prise en main l'applicatif - explication des fonctionnalités et de leurs usages - description des éléments non fonctionnels si présents

Livraison

Timing
projet rendu à la date prévu - l'estimation du temps de développement est juste

Sources nécessaires
les sources sont rendues en totalité - les sources sont exploitables

Choix de développement

Respect du sujet
développement des fonctionnalités demandées ou choix précis, justifié, suivi et complet d'un développement en rapport avec le sujet