TP - COMPTAGE DE CARACTÈRES - ARITHMÉTIQUE SUR LES CODES ASCII

Le but de ce TP est de réaliser un programme permettant de lire puis d'analyser une suite de caractères saisie par l'utilisateur. Il permet notamment de montrer les opérations réalisables sur des caractères. Il permet aussi de visualiser les codes ASCII (http://en.wikipedia.org/wiki/ASCII) utilisés pour représenter les caractères en mémoire.

Comptage de caractères

Il s'agit ici de faire un programme qui permet de lire une suite de caractères au clavier puis de les analyser au fur à mesure. La suite est terminée lorsque l'utilisateur tape un point (`'.'`).

Les caractères seront lus un par un en utilisant une seule variable.

L'analyse demandée consiste à compter, puis à afficher en fin de programme :

- le nombre d'occurrences des 10 chiffres (i.e. le nombre de chiffres de `'0'` à `'9'`)
- le nombre de caractères alphabétiques
- le nombre total de caractères d'espacement : espace (`' '`), tabulation (`'\t'`) et retour à la ligne (`'\n'`)

Préparation

1. Préparer des exemples en complétant les colonnes non grisées du tableau ci-dessous.

Valeurs fournies au clavier (Donner une liste d'au moins 10 caractères dans chacune des cases ci-dessous)	Valeur que devrait afficher le Programme	Résultat affiché après exécution	Le fonctionnement est-il conforme ?
Bonjour 1001 pattes et 106 voitures09.	5 espacements 23 caractères alpha 9 chiffres ...	Il y a 9 chiffres Il y a 23 lettres Il y a 42 espacements	non
1 test tab.	1 chiffre 1 espacement 7 caractères	Il y a 1 chiffres Il y a 7 lettres Il y a 13 espacements	non
1 autre test afin de tester 456.	4 chiffres 6 espacements 21 Caractères	Il y a 4 chiffres Il y a 21 lettres Il y a 36 espacements	non

2. Faire la vue externe, en déduire le prototype puis écrire l'algorithme de chacun des sous-programmes suivants qui réalise un test sur un caractère et retourne un booléen :

- `estUnChiffre()`
- `estUneLettre()`
- `estUnEspacement()`

3. Écrire le programme principal. Comme indiqué précédemment, chaque caractère doit être lu puis analysé par les sous-programmes dans le but d'incrémenter les compteurs.

Réalisation en séance

1. Ouvrir, dans **Netbeans**, un nouveau projet `TP04`. Copier le fichier `TLP_CPP\modele.cpp` en le renommant `cpt_char.cpp` dans le réportoire `TP04`. Mettre dans le projet le fichier source : `cpt_char.cpp`
2. Traduire l'algorithme des sous-programmes et du programme principal en C++. Lors de l'édition dans **Netbeans**, la lecture des caractères dans le programme principal doit se faire à l'aide de la syntaxe suivante pour comptabiliser les caractères d'espacement.
   `cin >> noskipws >> caract;`
3. Pour tester le programme, il faut **valider individuellement chacun des 3 sous-programmes**. Pour cela vous utiliserez les jeux de test préparés précédemment et vous remplirez le tableau (cases grisées) pour comparer les résultats attendus et ceux que vous avez.
4. Corriger le code si nécessaire puis conclure.

Code ASCII et transformation de caractères

Dans cette partie, on étudie les codes ASCII des caractères en ajoutant deux fonctionnalités au programme de la partie précédente.

La **première fonctionnalité** consiste à afficher une variable de type caractère dans différents formats :

- Le caractère
- Le Code ASCII en hexadécimal (base 16)
- Le Code ASCII en décimal (base 10)

La bibliothèque `iostream` du langage C++ permet un affichage dans différentes bases.

Pour afficher un entier en hexadécimal on peut écrire :

```
cout << hex << "0x" << monEntier; // Si monEntier=195, le programme affiche 0xc3
```

Pour afficher un entier en décimal on peut écrire :

```
cout << dec << monEntier;
```

La **deuxième fonctionnalité** consiste à transformer un caractère alphabétique en minuscule (s'il s'agit d'une majuscule), et inversement. On doit s'aider des codes ASCII

fournis dans le tableau ci-dessous pour déduire comment tester si un caractère est en majuscule ou en minuscule.

Caractères	Codes en hexadécimal	Codes en décimal
'a' jusqu'à **'z'**	**0x61** jusqu'à **0x7a**	**97** jusqu'à **122**
'A' jusqu'à **'Z'**	**0x41** jusqu'à **0x5a**	**65** jusqu'à **90**

En reprenant le programme de la première partie du TP, on effectuera pour chaque caractère alphabétique la transformation des majuscules en minuscules et vice versa. Quel que soit le caractère, on fera un affichage de l'encodage : valeur hexadécimale et décimale du caractère.

Préparation

1. Préparer des exemples en complétant les colonnes non grisées du tableau ci-dessous.

Valeurs fournies au clavier	Valeur que devrait afficher le Programme	Résultat affiché après exécution	Le fonctionnement est-il conforme ?
C'est 1 Test.	c 0x63 99 ' 0x27 39 E 0x45 69 ... 2 espacements 8 caract alpha 2 espacements		
test	En hexadecimale : 0x74 En hexadecimale : 0x65 En hexadecimale : 0x73 En hexadecimale : 0x74 Il y a 0 chiffres Il y a 4 lettres Il y a 0 espacements	Le caractere e a pour code ASCII : En hexadecimale : 0x65 En decimale : 101 Le caractere e est une minuscule Le caractere e en majuscule est : E ... Il y a 0 chiffres Il y a 4 lettres Il y a 4 espacements	oui (en dehors des espacements)
2 n TsEst	Il y a 1 chiffres Il y a 6 lettres Il y a 2 espacements	... Il y a 1 chiffres Il y a 6 lettres Il y a 11 espacements	//

2. Faire la vue externe puis écrire en notation algorithmique chacun des sous-programmes suivants :
    - `afficheCodeCaractere()` : affiche un caractère passé en paramètre formel dans les 3 formats. Pour afficher les valeurs hexadécimale et décimale du caractère, il faut affecter le caractère à une variable de type entier (ce qui provoque l'affectation du code ASCII du caractère à l'entier).
    - `estUneMajuscule()`
    - `minus2majusc()`
    - `majusc2minusc()`
3. Écrire une nouvelle version du programme principal précédent en intégrant les appels aux nouveaux sous-programmes.

Réalisation

1. Copier le fichier `TLP_CPP\TP04\cpt_char.cpp` dans le dossier `TLP_CPP\TP04` en le renommant `transformation_char.cpp` puis ajouter ce nouveau fichier comme fichier source du projet de **Netbeans**,
2. Traduire l'algorithme en C++ et l'éditer dans **Netbeans**.
3. Tester le programme. Pour cela vous utiliserez les exemples d'utilisation préparés précédemment et vous remplirez le tableau (cases grisées) pour comparer les résultats attendus et ceux que vous avez.
4. Corriger le code si nécessaire puis conclure.

Utilisation des bibliothèques standards

Dans cette partie nous allons reprendre les fonctions que vous avez créées précédemment en utilisant des bibliothèques standards du langage C. Vous remplacerez les fonctions que vous avez développées par les fonctions fournies avec les bibliothèques.

Voici une liste de quelques fonctions disponibles : `isalpha()`, `isalnum()`, `isdigit()`, `isspace()`, `isupper()`, `islower()`, `tolower()`, `toupper()`, etc.

La documentation sur ces fonctions est disponible sous plusieurs formes :

- Accès en ligne : http://www.cplusplus.com/reference/cctype/

Préparation

1. Lire la documentation et donner le prototype de la fonction `isdigit()`. Quel est le type du paramètre formel en entrée ? Quel est le type de la valeur retournée par la fonction ?
2. Quelle est la valeur retournée par la fonction en `isdigit()` en cas d'échec (i.e. si la valeur donnée en entrée ne correspond pas à un chiffre) ?
3. Lire la documentation et donner le prototype de la fonction `tolower()`. Quel est le type du paramètre formel en entrée ? Quel est le type de la valeur retournée par la fonction ?
4. Quelle est la valeur retournée par la fonction en `tolower()` en cas d'échec (i.e. si la valeur donnée en entrée ne peut pas être transformée en minuscule) ?

Réalisation

1. Copier votre fichier source dans le dossier `TLP_CPP\TP04` en le renommant `transformlib_char.cpp` puis ajouter ce nouveau fichier comme fichier source du projet de **Netbeans**,
2. Éditer le fichier source dans**Netbeans**. Remplacer tous les appels de sous-programmes de traitement des caractères développés précédemment par ceux fournis avec les bibliothèques standards.
3. Tester le programme. Pour cela vous utiliserez à nouveau les exemples d'utilisation préparés précédemment et vous remplirez le tableau (cases grisées) pour comparer les résultats attendus et ceux que vous avez.
4. Corriger le code si nécessaire puis conclure.