Travail écrit no. 1

(Durée : 2 périodes)

NE RIEN INSCRIRE SUR LA DONNEE

Problème 1 (0.5 point)

1) Complétez la partie *<à compléter>* ci-dessous (et uniquement cette partie !) :

```
char* strncpy (char* to, const char* from, size_t size) {
   char* tmp = to;
   while (<à compléter>);   (size--&& (*to++ = *(from++)) != '\0')
   return tmp;
}
```

Pour rappel, la sémantique de la fonction *strncpy* est la suivante :

Recopie les *size* premiers caractères de la chaîne *from* dans la chaîne *to* et retourne un pointeur sur *to*. De manière plus précise :

- Si *size* <= *strlen(from)*, *size* caractères sont copiés dans *to*.
  Attention : Aucune marque de fin de chaîne ('\0') n'est copiée dans *to*
- Si *size* > *strlen(from)*, *strlen(from)* caractères ainsi qu'un '\0' sont copiés dans *to*.

2) Soient maintenant les 3 extraits de code (indépendants les uns des autres) suivants:

a)
```
char* from = (char*) "12";
char to[] = "ABC";
strncpy(to, from, 4);
```

b)
```
const char* from = "12";
char* to = "ABC";
strncpy(to, from, 3);
```

c)
```
char s[] = "AB";
char* to = s;
strncpy(to, "12", 2);
```

Indiquez lequel (lesquels) de ces extraits n'est (ne sont) pas correct(s) et pourquoi.

Problème 2 (1.6 point)

1) La fonction ci-dessous est censée inverser la chaîne de caractères *s* passée en paramètre :

```
 1 char* inverser (const char* s) {
 2   if (s) {          (rabbat)
 3      char* r = new char[strlen(s)];
 4      if (r) {
 5         char* ptr = r + strlen(s);
 6         *ptr = '\0';
 7         for(; *s; s++)
 8            --*ptr = *s;   *--ptr = *s;
 9      }   (char*)
10      return r;
11   }
12   return s;
13 }
```

Cette fonction n'est toutefois pas correcte.
Indiquez le no de la ligne (des lignes) fautive(s) et proposez un (des) correctif(s).

IMPORTANT
- La ligne no 1 ne doit en aucun cas être modifiée
- Aucune ligne de code ne doit être ajoutée ou supprimée

2) Rappel: La fonction *strncat* de *cstring*, dont le prototype est :

```
char* strncat(char* to, const char* from, size_t size);
```

concatène la chaîne *from* à la suite de la chaîne *to* et retourne un pointeur sur *to*... mais au plus *size* caractères de la chaîne *from* sont copiés. Ici, contrairement à *strncpy*, une marque de fin de chaîne est toujours ajoutée à la fin de la chaîne (quel que soit le critère qui arrête la concaténation).

Soit maintenant la proposition d'implémentation de *strncat* suivante :

```
 1 char* strncat (char* to, const char* from, size_t size) {
 2    char* tmp = to;
 3    if (size) {
 4       while (*to++);        while(*to) to++; // avance jusqu'à '\0'
 5       while (*to++ == *from++) {
 6          if (--size == 0) {
 7             to = '\n';
 8             break;
 9          }
10       }
11    }        tmp
12    return to;
13 }
```

Problème 3 (1.6 point)

Soit le code (incomplet) suivant :

```
<à compléter 1>

int main()
{
   srand(time(NULL));
   uint nAleatoire = uint(rand() % 2); // n vaudra 0 ou 1

   Direction direction = Direction::NORD;
   for (uint n = 0; n < 6; ++n)
      cout << (nAleatoire ? direction++ : ++direction) + n << " ";
   cout << endl;

   return EXIT_SUCCESS;
}

<à compléter 2>
```

Complétez, le plus proprement possible, les parties ***<à compléter>*** du code ci-dessus, de telle sorte que celui-ci affiche à l'exécution

- quand nAleatoire vaut 0 :
  `Est Ouest Est Ouest Est Ouest`
- quand nAleatoire vaut 1 :
  `Nord Sud Nord Sud Nord Sud`

IMPORTANT

- Le code de la fonction *main* ne doit en aucun cas être modifié
- Implémentez le type *Direction* en tant que type énuméré **<u>fortement typé</u>**

Cette implémentation n'est toutefois pas correcte.
Indiquez le no de la ligne (des lignes) fautive(s) et proposez un (des) correctifs.

IMPORTANT

- La ligne no 1 ne doit en aucun cas être modifiée
- Aucune ligne de code ne doit être ajoutée ou supprimée
- Aucune fonction prédéfinie de la librairie standard ne doit être utilisée

3) Proposez une implémentation de la fonction *adrValMin* dont la sémantique est la suivante :
adrValMin est une fonction sans valeur de retour qui prend en paramètre d'entrée un tableau (classique) de *int* et qui renvoie en paramètre de sortie l'**adresse** de la plus petite valeur du tableau (*NULL* si le tableau est vide).

IMPORTANT

- Aucune variable locale ne doit être déclarée dans la fonction
- Aucune fonction prédéfinie de la librairie standard ne doit être utilisée

```
void adrValMin( int tab[], const int **adr, const int taille)
{
    if (taille)
    {
      *adr = tab[0];
      for (register unsigned int i=0; i<taille; i++)
        if( tab[i] < **adr )
            *adr = (tab+i)   ou &tab[i]
    }
    else *adr = nullptr
}
```

Problème 4 (1.3 point)

1) Concevoir, de la manière la plus propre et la plus évolutive possible, **la spécification d'une librairie** permettant de modéliser des singes conformément au cahier des charges suivant :

   - Un singe a un nom, est soigné par un ou plusieurs soigneurs (dont seul le nom nous intéresse) et est soit un gorille, soit un chimpanzé
   - Si le singe est un chimpanzé, on souhaite enregistrer en plus son âge et s'il est agressif ou non
   - Si le singe est un gorille, on souhaite enregistrer en plus son poids (réel exprimé en [kg])

   IMPORTANT
   - Pour les chaînes de caractères, considérer le type *string*
   - Il n'est PAS demandé de proposer quelque fonction que ce soit (constructeur, fonction membre ou non) dans la spécification demandée

2) En supposant le point 1) résolu et que le compilateur utilisé supporte la norme C++ 2011, déclarez, **de la manière la plus concise possible**, les 2 singes suivants:

   - Chita : un chimpanzé de 10 ans, non agressif, soigné par Pierre
   - King Kong : un gorille de 300 kg soigné par Paul et Jacques

Correction TE 1.
INFO

P.1

1. `while (size-- && (*to++ = *from++) != '\0')`

   est sous-entendu.

2. Le deuxième est incorrecte car to pointe sur une chaîne constante non-modifiable

   `char* to = "ABC"` Warning à la compilation.

   ↑ constant

Problème 2.

1.
```
new (nothrow) char[(strlen(s)+1]
    *--ptr = *s
return (char*) s
```

2.
```
while (*to) to++;
while (*to++ = *from++)
{
  if (--size == 0)
  {
    to = '\0';
    break;
  }
}
return temp;
```

3.
```
void adrValMin (const int tab[], const int taille, (const)int **adr)
{
  if (taille == 0)
    *adr = Null; /ou nullptr;
  else
  {
    *adr = tab  //ou &tab[0]
    for (int i = 1; i < taille; ++i)
    {  if (tab[i] < **adr)
       {
         *adr = tab+i; //ou &tab[i]
       }
    }
  }
}
```

Problème 3.

```
#include
typedef unsigned int uint;
enum class Direction {Nord, Est, Sud, Ouest};
const string DIRECTIONS[] = {"Nord", "Est", "Sud", "Ouest"};
const uint NB_Directions = sizeof(DIRECTIONS / sizeof(string));

ostream& operator <<(ostream&, const Direction&);
Direction operator + (const Direction&, const uint);
Direction operator ++ (Direction&, int);
Direction operator ++ (Direction&);


ostream& operator <<(ostream& os, const Direction& d)
{
    return os << DIRECTIONS[(int)d];
}
Direction operator +(const Direction& d, const uint n)
{
  return Direction(((int)d + n) % NB_DIRECTIONS);
}
Direction operator ++ (Direction& d, int n)
{
    Direction temp = d;
    d = d+1;
    return temp;
}
Direction operator ++ (Direction& d)
{
    return d = d+1;
}
```

#ifndef
#define ...

Problème 4

1)
```
#include <string>
#include <vector>
using namespace std;

typedef unsigned short ushort;
typedef string seigneur
typedef vector<seigneur> Seigneurs;
enum class Espece {Chimpanze, Gorille};

struct chimpanze
{
    ushort age;
    bool est agressif;
};
struct Gorille
{
    double poids;
};
union Specificites
{
    Chimpanze chimpanze;
    Gorille gorille;
};
```

Union contenant des structures

On ne peut initialiser que le 1er champ avec un {...}

```
struct Singe
{
    string nom;
    Seigneurs seigneurs;
    Espece espece;
    Specificites specificites;
};
#endif
```

seigneurs : vector<string>, seigneurs : string

enum class Espece {chimpanze, gorille}

variante Singe chita = { ... } ou chita ({ })

" chita { };

2)
```
Singe
    chita {"chita", {"Pierre"}, Espece::Chimpanze, {{10, false}}},
    KingKong {"KingKong", {"Paul", "Jacques"}, Espece::Gorille},

    KingKong.specificites.gorille.poids = 300.
```

ici chimpanzé

on ne peut initialiser que le premier champ de l'union avec des agrégats d'où la ligne pour KingKong