UNIVERSITÉ DE PAU ET DES PAYS DE L'ADOUR | COLLÈGE STEE SCIENCES ET TECHNOLOGIES POUR L'ÉNERGIE ET L'ENVIRONNEMENT

Techniques de programmation - Projet - BM

Samson Pierre <samson.pierre@univ-pau.fr>

18/11/2021

Vous devez créer un programme nommé BM (Base Manager) qui doit lire un fichier XML dont le chemin est passé en ligne de commande. Ce fichier XML contient des données concernant une base constituée d'installations. Ce programme doit proposer de saisir des commandes pour afficher des informations contenues dans ce fichier XML.

Le travail à rendre

L'archive à rendre doit contenir exactement ces fichiers :

```
$ tar tf projet-jean-bonbeurre-samson-pierre.tar.xz | sort
projet-jean-bonbeurre-samson-pierre/
projet-jean-bonbeurre-samson-pierre/base-1.xml
projet-jean-bonbeurre-samson-pierre/base.c
projet-jean-bonbeurre-samson-pierre/base.h
projet-jean-bonbeurre-samson-pierre/bm.c
projet-jean-bonbeurre-samson-pierre/facility.c
projet-jean-bonbeurre-samson-pierre/facility.h
projet-jean-bonbeurre-samson-pierre/Makefile
$
```

Les prénoms `jean` et `samson` ainsi que les noms `bonbeurre` et `pierre` sont à remplacer par les vôtres. Le fichier `Makefile` contient les règles dont se sert le programme `make`. Les fichiers `base.c` et `base.h` contiennent respectivement les définitions et les déclarations pour une base. Les fichiers `facility.c` et `facility.h` contiennent respectivement les définitions et les déclarations pour une installation. Le fichier `bm.c` contient les définitions pour le programme (notamment celle de la fonction principale). Le fichier `base-1.xml` contient des données concernant une base.

La compilation du projet doit se passer ainsi :

```
$ make
gcc -std=c89 -pedantic -Wall -Werror -g `pkg-config libxml-2.0 --cflags` -c -o base.o base.c
gcc -std=c89 -pedantic -Wall -Werror -g `pkg-config libxml-2.0 --cflags` -c -o facility.o facility.c
gcc -std=c89 -pedantic -Wall -Werror -g `pkg-config libxml-2.0 --cflags` -o bm.out bm.c base.o
facility.o `pkg-config libxml-2.0 --libs-only-L` `pkg-config libxml-2.0 --libs-only-l`
$
```

Vous êtes autorisés à utiliser les fonctions suivantes :

```
char *fgets(char *s, int size, FILE *stream);
int fprintf(FILE *stream, const char *format, ...);
void free(void *ptr);
int getchar(void);
void *malloc(size_t size);
int printf(const char *format, ...);
void *realloc(void *ptr, size_t size);
int strcmp(const char *s1, const char *s2);
size_t strlen(const char *s);
char *strstr(const char *haystack, const char *needle);
double strtod(const char *nptr, char **endptr);
long int strtol(const char *nptr, char **endptr, int base);
void xmlCleanupParser(void);
xmlNodePtr xmlDocGetRootElement(const xmlDoc *doc);
void xmlFreeDoc(xmlDocPtr cur);
xmlChar *xmlGetProp(const xmlNode *node, const xmlChar *name);
xmlChar *xmlNodeListGetString(xmlDocPtr doc, const xmlNode *list, int inLine);
xmlDocPtr xmlParseFile(const char *filename);
int xmlStrcmp(const xmlChar *str1, const xmlChar *str2);
```

Pour obtenir une bonne note, vous devez respecter ces règles :

— les fichiers doivent être encodés en UTF-8
— les fichiers ne doivent pas contenir de fautes d'orthographe
— les noms de fichiers doivent être ceux indiqués dans ce sujet

— les fichiers d'en-tête doivent être identiques à ceux du sujet
— le fichier XML doit être identique à celui du sujet
— les options de compilation doivent être celles précisées dans ce sujet
— les affichages à l'écran doivent correspondre à ceux du sujet
— la solution doit se rapprocher au maximum de ce qui est demandé dans ce sujet
— la solution doit ignorer tout noeud ou attribut qui ne décrit pas une base constituée d'installations
— la solution doit accepter tout fichier XML similaire à celui du sujet (`base-1.xml` n'est qu'un exemple)
— le code doit être correctement indenté
— le code doit être homogène concernant le nom des variables, les espaces, ...
— la mémoire allouée doit être correctement libérée
— les fonctions que vous utilisez dans votre code doivent figurer parmi celles autorisées dans ce sujet
— les valeurs de retour des fonctions pouvant échouer doivent être vérifiées afin de traiter les erreurs
— les messages d'erreur doivent être envoyés dans le flux d'erreur standard
— les autres messages seront envoyés dans le flux de sortie standard
— le code retourné par un processus rencontrant une erreur doit être 1
— le code retourné par un processus se terminant normalement doit être 0
— l'archive ne doit pas contenir d'autres fichiers que ceux demandés dans ce sujet
— l'archive doit être envoyée au plus tard le 10/12/2021 à 23:59
— l'archive doit être envoyée par e-mail à l'adresse `samson.pierre@univ-pau.fr`
— le sujet de l'e-mail doit être `TDP - Projet - Jean Bonbeurre - Samson Pierre`
— les prénoms `Jean` et `Samson` sont à remplacer par les vôtres
— les noms `Bonbeurre` et `Pierre` sont à remplacer par les vôtres
— le travail est à réaliser impérativement en binôme
— le binôme doit travailler en collaboration via un projet privé sur la forge `https://git.univ-pau.fr/`
— tout travail similaire à un autre sera sanctionné par une note nulle (0/20)

Le programme

Le programme doit lire un fichier XML dont le chemin est passé en ligne de commande. Si le nombre de paramètres passés en ligne de commande est différent de un, le programme doit afficher un message d'erreur et quitter. Idem si l'analyse du fichier échoue (exemple : s'il n'existe pas).

```
$ ./bm.out
./bm.out: Invalid number of arguments
$ ./bm.out file.xml
I/O warning : failed to load external entity "file.xml"
./bm.out: Unable to parse the document
$
```

Le programme doit proposer de saisir des commandes :

```
$ ./bm.out base-1.xml
BM>
```

La commande `h` permet d'obtenir cet affichage :

```
$ ./bm.out base-1.xml
BM> h
b: Prints the base
c: Prints the base country
d: Prints the base date
f: Prints the base facilities
fc COST: Prints the base facilities with the cost equal to COST
fcge COST: Prints the base facilities with the cost greater than or equal to COST
fcgt COST: Prints the base facilities with the cost greater than COST
fcle COST: Prints the base facilities with the cost less than or equal to COST
fclt COST: Prints the base facilities with the cost less than COST
fn NAME: Prints the base facilities with the name containing NAME
h: Prints this help
n: Prints the base name
t: Prints the base total cost
v: Prints the BM version
q: Quits BM
BM>
```

La commande `v` permet d'obtenir cet affichage :

```
$ ./bm.out base-1.xml
BM> v
BM (Base Manager) 20211114
```

```
4
5  Copyright (C) 2021 Jean Bonbeurre and Samson Pierre.
6
7  Written by Jean Bonbeurre <jean.bonbeurre@univ-pau.fr> and Samson Pierre
   <samson.pierre@univ-pau.fr>.
8  BM>
```

Les prénoms `Jean` et `Samson`, les noms `Bonbeurre` et `Pierre` ainsi que les adresses e-mail `jean.bonbeurre@univ-pau.fr` et `samson.pierre@univ-pau.fr` sont à remplacer par les vôtres. La version `20211104` est à remplacer par la version de votre programme.

La commande `q` permet de quitter le programme :

```
1  $ ./bm.out base-1.xml
2  BM> q
3  Goodbye!
4  $
```

Les autres commandes affichent un contenu spécifique à la base :

```
1  $ ./bm.out base-1.xml
2  BM> b
3  base 1 (2022-09-01, france)
4  BM> c
5  france
6  BM> d
7  2022-09-01
8  BM> f
9  hangar 1, 80 m2, 490000.99 USD
10 hangar 2, 80 m2, 490000.99 USD
11 laboratory, 20 m2, 290000.50 USD
12 living quarters, 20 m2, 190000.00 USD
13 workshop, 20 m2, 290000.50 USD
14 BM> fc 290000.5
15 laboratory, 20 m2, 290000.50 USD
16 workshop, 20 m2, 290000.50 USD
17 BM> fcge 290000.5
18 hangar 1, 80 m2, 490000.99 USD
19 hangar 2, 80 m2, 490000.99 USD
20 laboratory, 20 m2, 290000.50 USD
21 workshop, 20 m2, 290000.50 USD
22 BM> fcgt 290000.5
23 hangar 1, 80 m2, 490000.99 USD
24 hangar 2, 80 m2, 490000.99 USD
25 BM> fcle 290000.5
26 laboratory, 20 m2, 290000.50 USD
27 living quarters, 20 m2, 190000.00 USD
28 workshop, 20 m2, 290000.50 USD
29 BM> fclt 290000.5
30 living quarters, 20 m2, 190000.00 USD
31 BM> fn or
32 laboratory, 20 m2, 290000.50 USD
33 workshop, 20 m2, 290000.50 USD
34 BM> n
35 base 1
36 BM> t
37 1750002.98 USD
38 BM>
```

Si la commande saisie n'existe pas, le programme doit afficher un message d'erreur. Idem si le paramètre de la commande manque, si le paramètre de la commande est incorrect ou si la commande dépasse 18 caractères :

```
1  $ ./bm.out base-1.xml
2  BM> hello
3  ./bm.out: Invalid command
4  BM> fc
5  ./bm.out: Missing parameter for the fc command
6  BM> fc abc
7  ./bm.out: Invalid parameter for the fc command
8  BM> fc 0123456789012345
9  ./bm.out: Too many characters for the command
10 BM>
```

Les fichiers d'en-tête

Voici le fichier d'en-tête `base.h` :

```c
/**
 * \file base.h
 */
#ifndef BASE_H
#define BASE_H
#include "facility.h" /* for facility_t */
/**
 * A base.
 */
typedef struct
{
        char *country; /**< The base location (country). */
        int day; /**< The base date (day). */
        facility_t **facilities; /**< The base facilities. */
        int month; /**< The base date (month). */
        char *name; /**< The base name. */
        int nfacilities; /**< The base number of facilities. */
        int year; /**< The base date (year). */
} base_t;
/**
 * Adds a facility to a base.
 * \param base The base.
 * \param facility The facility.
 * \return -1 on error (i.e., if the memory allocation is a failure), else 0.
 */
int base_add_facility(base_t *base, facility_t *facility);
/**
 * Creates a base.
 * \return NULL on error (i.e., if the memory allocation is a failure), else a base.
 */
base_t *base_create();
/**
 * Frees a base.
 * \param base The base.
 */
void base_free(base_t *base);
/**
 * Handles the b command for a base.
 * \param base The base.
 */
void base_handle_b(base_t base);
/**
 * Handles the c command for a base.
 * \param base The base.
 */
void base_handle_c(base_t base);
/**
 * Handles the d command for a base.
 * \param base The base.
 */
void base_handle_d(base_t base);
/**
 * Handles the f command for a base.
 * \param base The base.
 */
void base_handle_f(base_t base);
/**
 * Handles the fc command for all the facilities of a base.
 * \param base The base.
 * \param cost The facility cost.
 */
void base_handle_fc(base_t base, double cost);
/**
 * Handles the fcge command for all the facilities of a base.
 * \param base The base.
 * \param cost The facility cost.
 */
void base_handle_fcge(base_t base, double cost);
/**
 * Handles the fcgt command for all the facilities of a base.
 * \param base The base.
 * \param cost The facility cost.
 */
void base_handle_fcgt(base_t base, double cost);
/**
 * Handles the fcle command for all the facilities of a base.
```

```
 * \param base The base.
 * \param cost The facility cost.
 */
void base_handle_fcle(base_t base, double cost);
/**
 * Handles the fclt command for all the facilities of a base.
 * \param base The base.
 * \param cost The facility cost.
 */
void base_handle_fclt(base_t base, double cost);
/**
 * Handles the fn command for all the facilities of a base.
 * \param base The base.
 * \param name The facility name.
 */
void base_handle_fn(base_t base, const char *name);
/**
 * Handles the n command for a base.
 * \param base The base.
 */
void base_handle_n(base_t base);
/**
 * Handles the t command for a base.
 * \param base The base.
 */
void base_handle_t(base_t base);
#endif
```

Voici le fichier d'en-tête `facility.h` :

```
/**
 * \file facility.h
 */
#ifndef FACILITY_H
#define FACILITY_H
/**
 * A facility.
 */
typedef struct
{
        int area; /**< The facility area (in m2). */
        double cost; /**< The facility cost (in USD). */
        char *name; /**< The facility name. */
} facility_t;
/**
 * Creates a facility.
 * \return NULL on error (i.e., if the memory allocation is a failure), else a facility.
 */
facility_t *facility_create();
/**
 * Frees a facility.
 * \param facility The facility.
 */
void facility_free(facility_t *facility);
/**
 * Handles the f command for a facility.
 * \param facility The facility.
 */
void facility_handle_f(facility_t facility);
/**
 * Handles the fc command for a facility.
 * \param facility The facility.
 * \param cost The facility cost.
 */
void facility_handle_fc(facility_t facility, double cost);
/**
 * Handles the fcge command for a facility.
 * \param facility The facility.
 * \param cost The facility cost.
 */
void facility_handle_fcge(facility_t facility, double cost);
/**
 * Handles the fcgt command for a facility.
 * \param facility The facility.
 * \param cost The facility cost.
 */
void facility_handle_fcgt(facility_t facility, double cost);
```

```
48 /**
49  * Handles the fcle command for a facility.
50  * \param facility The facility.
51  * \param cost The facility cost.
52  */
53 void facility_handle_fcle(facility_t facility, double cost);
54 /**
55  * Handles the fclt command for a facility.
56  * \param facility The facility.
57  * \param cost The facility cost.
58  */
59 void facility_handle_fclt(facility_t facility, double cost);
60 /**
61  * Handles the fn command for a facility.
62  * \param facility The facility.
63  * \param name The facility name.
64  */
65 void facility_handle_fn(facility_t facility, const char *name);
66 #endif
```

Le fichier XML

Voici le fichier XML `base-1.xml` :

```
1  <base name="base 1">
2          <date>
3                  <day>1</day>
4                  <month>9</month>
5                  <year>2022</year>
6          </date>
7          <country>france</country>
8          <facilities>
9                  <facility name="hangar 1">
10                         <area>80</area>
11                         <cost>490000.99</cost>
12                 </facility>
13                 <facility name="hangar 2">
14                         <area>80</area>
15                         <cost>490000.99</cost>
16                 </facility>
17                 <facility name="laboratory">
18                         <area>20</area>
19                         <cost>290000.5</cost>
20                 </facility>
21                 <facility name="living quarters">
22                         <area>20</area>
23                         <cost>190000</cost>
24                 </facility>
25                 <facility name="workshop">
26                         <area>20</area>
27                         <cost>290000.5</cost>
28                 </facility>
29         </facilities>
30 </base>
```