UNIVERSITETI I PRISHTINËS
HASAN PRISHTINA

FAKULTETI I INXHINIERISË
ELEKTRIKE DHE KOMPJUTERIKE

PROGRAMIMI SISTEMOR

Detyra e parë

ABSTRAKTI

Temat që do të shtjellohen në detyrën e parë janë: i) pointerët, ii) strukturat, iii) file-at, iv) stringjet, v) serializimi, dhe vi) procesimi i opsionëve dhe argumentëve të aplikacionit.

AFATI I DORËZIMIT TË DETYRËS

E premte, më 02.06.2019

ARSIMTARI I LËNDËS

ASS. PROF. DR. IDRIZ SMAILI

PRISHTINË, MAI 2019

PËRMBAJTJA

LISTA E FIGURAVE

ORGANIZIMI

Detyra e parë ka të bëjë me këtë tematikë:

- **Pointerët**: aritmetika e pointerëve, call-by-reference, etj.,
- **Strukturat**: definimi i strukturave, instanciimi i objekteve nga strukturat, pointerët në struktura, dereferencimi i pointerëve në struktura, etj.,
- **File**-at: krijimi dhe hapja dhe manipulimi me file-a (posaqërisht ata binarë), serializimi i strukturave komplekse të të dhënave (p.sh. container objects), dhe
- **Opsion**-et: procesimi i opsionëve dhe argumentëve të aplikacionëve.

Detyra konsiderohet e kryerë nëse i) janë të implementuara të gjitha kërkesat (funksionale dhe jofunksionale), ii) nëse është dokumentuar secili funksion (public dhe "private/protected" ose static në gjuhën programuese C), iii) është bërë përshkrimi se si duhet të përdoret/testohet aplikacioni (detyra), si dhe iv) detyra është mbrojtur me sukses para arsimtarit të lëndës[1].

Numri maksimal i pikëve nga absolvimi me sukses është **pesëmbëdhjetë** (**15**). Në rast se nuk plotësohen kërkesat jo funksionale, atëherë do të ketë pikë negative si vijon:

- -2 pikë: ***nFncReq 1***,

[1] Skype meetings do të përcaktohen nga arsimtari i lëndës.

- -2 pikë: ***nFncReq 2***, dhe
- -2 pikë: ***nFncReq 3***.

Në rast se detyra e dorëzuar nuk është e saktë, që d.t.th. se njëra nga kërkesat funksionale nuk është plotësuar, atëherë detyra duhet dorëzuar përsëri, mirëpo numri maksimal i pikëve në këtë rast do të dekrementohet për **tre** (**3**), që d.t.th. që numri maksimal i pikëve pas iteracionit të parë është **dymbëdhjetë** (**12**).

FILE STRUKTURA

File struktura e detyrës duhet të jetë e ngjajshme me file strukturën e shembullit të parë (exercise_1) gjatë ligjëratave, që d.t.th. që në root directoriumin e detyrës duhet ketë vetëm tre direktoriume, `Makefile` dhe `readme.txt`. Direktoriumet e lejuara janë:

- **`src`**: në këtë direktorium duhet të ruhen vetëm source file-at, q.d.th. *.c,
- **`include`**: në këtë direktorium duhet të ruhen vetëm include file-at,
- **`doc`**: file-i i vetëm që duhet të ruhet në këtë direktorium është exc1.doc, i cili përdorët për konfigurim të `doxygen`-it.

Makefile duhet të ndërtohet asisoi që gjatë procesit të make të krijohet një direktorium **`output`**, në të cilin do të krijohen këto direktoriume:

- **`build`**: në këtë direktorium do të gjenerohen object files (*.o), të cilët krijohen nga gcc compiler-i gjatë compile-imit të source file-ave,
- **`exe`**: në këtë direktorium do të gjenerohet executable pasi të jenë linkuar të gjithë object files, dhe
- **`docu`**: është direktoriumi në të cilin do të gjenerohet online dokumentacioni nga `doxygen` tool.

Direktoriumet e mësipërme do të krijohen gjatë make procesit pëmes komandave:

```
make all
make dox
```

Në file-in `readme.txt` duhet ta përshkruani me anë të një paragrafi aplikacionin (detyrën) si dhe t'i specifikoni hapat që shfrytëzuesi duhet t'i bëjë në mënyrë që të mund ta testoj aplikacionin e juaj.

DORËZIMI I DETYRËS

Detyra duhet të dorëzohet përmes emailit më së largu gjerë të premten e fundit të muajit maj, më **02.06.2019** para orës **24:00**. Detyrat e dorëzuara me vonësë nuk do të merren parasysh, dhe numri i pikëve do të konsiderohet i barabartë me **zero**.

Është shumë me rëndësi të cekët që para dorëzimit të detyrës, ju duhet patjeter t'i pastroni direktoriumet e gjeneruara përmes komandës:

```
make clean
```

File-i me këtë emërtim `emri_mbiemri_detyra1.tgz`. duhet t'i bashkangjitt emaili gjatë dorëzimi të detyrës. Krijimi i këtij file-i bëhet përmes këtyre komandave:

```
cd root_exercise_dir
make clean
tar czf ~/emri_mbiemri_detyra1.tgz detyra1
```

SPECIFIKACIONI

KËRKESAT FUNKSIONALE

FncReq 1: Të implementohet sistemi softuerik që mundëson menaxhimin e listës së studentëve që ndjekin lëndën.

FncReq 2: Sistemi duhet të mundësoi mbledhjen e këtyre shënimeve për secilin student[2]: i) emrin, ii) mbiemrin, iii) nr. e index-it, iv) moshën (int), dhe v) adresën.

FncReq 3: Sistemi duhet të kufizoi në dyzet (40) numrin maksimal të studentëve që menaxhohen.

FncReq 4: Sistemi duhet të përdor file-a binar për ruajtjen e shënimeve për studentët e menaxhuar.

FncReq 5: File-i duhet të hapet gjatë inicializimit të applikacionit dhe të mbyllet gjatë fazës së përfundimit të aplikacionit.

FncReq 6: Sistemi duhet të ketë synopsis:

```
SYNOPSIS

   exc1 [OPTION] ...
    -f arg,  (mandatory) the file name
    -a,      (optional) append a string to the file
    -i,      (optional) start the interactive mode
    -l,      (optional) list all strings stored in the file
```

[2] Studentët duhet të deklarohen si objekte të user defined data structure përmes `struct` si `student_t`!

FncReq 7: Sistemi duhet përkrah dy lloje të opsioneve: i) një opsion obligativ, dhe ii) tre opsione jo-obligative.

FncReq 8: Sistemi duhet të kontrolloi se a është definuar opsioni obligativ.

FncReq 9: Sistemi duhet përkrah dy lloje të opsioneve: i) me argumente, dhe ii) pa argumente.

FncReq 10: Sistemi duhet të kontrolloi se a është definuar argumenti i opsionit me argument.

FncReq 11: Sistemi duhet t'i ofroi shfrytëzuesit dy mode të punës: i) interaktiv, dhe ii) jo-interaktiv. Opsionet: i) `-a`, dhe ii) `-l`, i takojnë modit të punë jo-interaktiv, ndërsa opsioni `-i` i takon modit interkativ.

FncReq 12: Sistemi duhet të kontrolloi që vetëm njëri mod i punës është zgjedhur nga shfrytëzuesi, d.t.th. interaktiv apo jo-interaktiv.

FncReq 13: Në rast se opsioni `-a` është specifikuar, atëherë sistemi duhet t'i mundësoi shfrytëzuesit regjistrimin e një studenti të ri në fund të file-it të specifikuar përmes opsionit `-f`.

```
/**************************************************************************//**
 * @brief Writes an input string into the file
 *
 * First the four (4) bytes will be written indicating the length of the
 * string to be written, and then the string itself will be written.
 *
 * @param[in,out] fp - file pointer
 * @param[in] str - the input string
 *
 * @retval 0  in case an error was occured
 * @retval >0 number of bytes written in the file
 *****************************************************************************/
int str_write (FILE *fp, const char *str)
{
  int status = 0;
  int length = 0;

  length = str_len (str);

  status = (int) fwrite ((const void *) &length, (size_t) 1,
                         (size_t) SER_INT_LEN, fp);

  if (status == 0)
    {
       printf ("\nError writing length of string '%s' to the file", str);
       return status;
    }

  status = (int) fwrite ((const void *) str, (size_t) 1,
                         (size_t) length, fp);

  if (status == 0)
    {
       printf ("\nError writing string '%s' to the file", str);
    }

  return status;
}
```

Figura 1: Shembulli i coding style si dhe i dokumentimit online përmes doxygen

FncReq_14: Në rast se opsioni `-l` është specifkuar, atëherë sistemi duhet t'i listoi të gjithë studentët e regjistruar në file-in e specifikuar përmes opsionit `-f`.

FncReq 15: Sistemi duhet t'i përkrah këto komanda: i) `append`, ii) `list`, si dhe iii) `quit`.

FncReq 16: Dy komandat e para (`append`, and `list`) duhet të egzekutojnë të njëjtin kod njësoi si kodi i cili egzekutohet nga opsionet jo-interaktive `-a` dhe `-l`.

FncReq 17: Kommanda `quit` përfundon aplikacionin.

FncReq 18: Para përfundimit të aplikacionit sistemi duhet t'i ruaj shënimet në file-in binar të specifikuar përmes opsionit `-f`.

FncReq 19: Sistemi duhet ta egzekutoi funksionin `quit`, në rast se shfrytëzuesi e përdor kombinimin e tasteve `Ctrl+C`[3].

KËRKESAT JOFUNKSIONALE

nFncReq 1: Coding style i paraqitur si në Figura 1 duhet të aplikohet gjatë implementimit të sistemit.

nFncReq 2: Dokumentimi online i paraqitur si në Figura 1 duhet të aplikohet gjatë implementimit të secilit funksion të sistemit. Dokumentimi i secilit funksion duhet të përmbaj: i) brief – një përshkrim i shkurtër i funksionit, ii) përshkrim i shkurtër i secilit argument të funksionit në formatin e paraqitur në figurën e mësipërme, si dhe iii) dokumentimi i vlerave kthyese të funksionit.

```
int str_write ( FILE *        fp,
                const char *  str
              )
```

Writes an input string into the file.

First the four (4) bytes will be written indicating the length of the string to be written, and then the string itself will be written.

Parameters

`[in,out]` **fp** - file pointer

`[in]` **str** - the input string

Return values

0 in case an error was occured

>0 number of bytes written in the file

Definition at line **50** of file **f_ser.c**.

Figura 2: Dokumentimi i funksionëve përmes doxygen

[3] Të implementohet signal-i SIGINT

nFncReq 3: Sistemi duhet të egzekutohet *free of memory leaks*. Kontrolli për *memory leaks* duhet të bëhet duke e shfrytëzuar tools-in `valgrind` dhe opsionin e aktivizuar `--track-origins=yes`. Sistemi konsiderohet *free of memory leaks* nëse rezultati përmban këtë fjali *„All heap blocks were freed -- no memory leaks are possible“*.

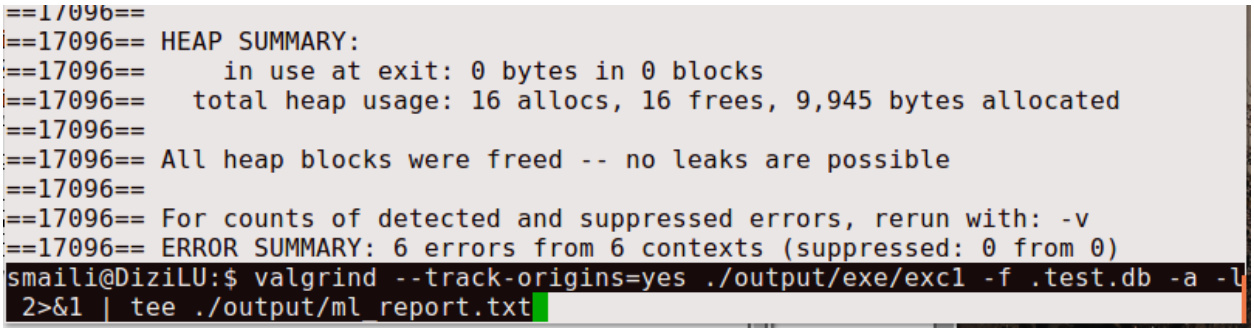

Figura 3: Kontrolli për memory leaks