



KOALA

B3- C++ Pool

B-PAV-242

Day 03

my string

v1.1



Day 03

my string

repository name: cpp_d03
repository rights: ramassage-tek
language: C
group size: 1



- Your repository must contain the totality of your source files, but no useless files (binary, temp files, obj files,...).
- All the bonus files (including a potential specific Makefile) should be in a directory named *bonus*.
- Error messages have to be written on the error output, and the program should then exit with the 84 error code (0 if there is no error).



If you do half the exercises because you have comprehension problems, it's okay, it happens. But if you do half the exercises because you're lazy, and leave at 2PM, you WILL have problems. Do NOT tempt the devil.



Every function implemented in a header, or unprotected header, leads to 0 to the exercise. Every class must possess a constructor and a destructor.



Every output goes to the standard output, and will be ended by a newline, unless specified otherwise.



Any use of "friend" will result in the grade -42 no questions asked.



To avoid compilation problems, please include necessary files within your headers.

Please note that none of your files must contain the main function except when explicitly asked. We will use our own main function to compile and test your code.



Exercise 00

my_string

repository subdir: /ex00
compilation: gcc -Wall -Wextra -Werror -std=gnu99
files to turn in: String.h, String.c
forbidden functions: none
points: 1

Create a **String** module that contains the following:

- a **char * str** member,
- an initialization function that assigns the **s** value to the **String** module instance:

```
void StringInit (String *this , char const *s);
```

- a destructor function that destroys a **String** module instance:

```
void StringDestroy (String *this);
```

Exercise 01

Assign

repository subdir: /ex01
compilation: gcc -Wall -Wextra -Werror -std=gnu99
files to turn in: String.h, String.c
forbidden functions: none
points: 1

Add the following two **assign** member functions.

```
void assign_s (String *this , String const *str);
```

The content of the **String** is equal to the content of the **String** passed as parameter.

```
void assign_c (String *this , char const *s);
```

The content of the **String** is equal to the content of the **char*** passed as parameter.



Reminder: Member functions can only be called from a **String** instance.
Remember to assign your function pointers.
Be careful not to leave any memory leaks.



Exercise 02

Append

repository subdir: /ex02
compilation: gcc -Wall -Wextra -Werror -std=gnu99
files to turn in: String.h, String.c
forbidden functions: none
points: 1

Add the following two **append** member functions.

```
void append_s(String *this, String const *ap);
```

Copies the content of **ap** to the end of the **String**.

```
void append_c(String *this, char const *aps);
```

Copies the content of **aps** to the end of the **String**.

Exercise 03

At

repository subdir: /ex03
compilation: gcc -Wall -Wextra -Werror -std=gnu99
files to turn in: String.h, String.c
forbidden functions: none
points: 1

Add an **at** member function:

```
char at(String *this, size_t pos);
```

Returns the **char** that is at the **pos** position in our **String**.
If the position is incorrect, it returns -1.



Exercise 04

Clear

repository subdir: /ex04
compilation: gcc -Wall -Wextra -Werror -std=gnu99
files to turn in: String.h, String.c
forbidden functions: none
points: 1

Add a **clear** member function:

```
void clear(String *this);
```

Empties the **String**'s content.



Be careful with your pointers.

Exercise 05

Size

repository subdir: /ex05
compilation: gcc -Wall -Wextra -Werror -std=gnu99
files to turn in: String.h, String.c
forbidden functions: none
points: 1

Add a **size** member function:

```
int size(String *this);
```

Returns the size of the string.

If the string pointer is NULL, the function returns -1.



Exercise 06

Compare

repository subdir: /ex06
compilation: gcc -Wall -Wextra -Werror -std=gnu99
files to turn in: String.h, String.c
forbidden functions: none
points: 1

Add the following two **compare** member functions.

```
int compare_s(String *this, const String *str);
```

Compares the content of the **String** to **str**.
Results are the same as the **strcmp** libc function.

```
int compare_c(String *this, char const *str);
```

Compares the content of the **String** to **str**.
Results are the same as the **strcmp** function.

Exercise 07

Copy

repository subdir: /ex07
compilation: gcc -Wall -Wextra -Werror -std=gnu99
files to turn in: String.h, String.c
forbidden functions: none
points: 1

Add a **copy** member function:

```
size_t copy(String *this, char *s, size_t n, size_t pos);
```

Copies 'n' of the **String**'s characters in 's', starting from the **pos** position.
Returns the number of characters that have been copied.



Exercise 08

c_str

repository subdir: /ex08

compilation: gcc -Wall -Wextra -Werror -std=gnu99

files to turn in: String.h, String.c

forbidden functions: none

points: 1

Add a **c_str** member function.

```
char const *c_str( String *this );
```

Returns the buffer contained in the **String**.

Exercise 09

empty

repository subdir: /ex09

compilation: gcc -Wall -Wextra -Werror -std=gnu99

files to turn in: String.h, String.c

forbidden functions: none

points: 1

Add an **empty** member function:

```
int empty( String *this );
```

Returns 1 if the string is empty, and -1 otherwise.



Exercise 10

Find

repository subdir: /ex10
compilation: gcc -Wall -Wextra -Werror -std=gnu99
files to turn in: String.h, String.c
forbidden functions: none
points: 1

Add the following two **find** member functions.

```
int find_s(String* this, const String *str, size_t pos);
```

Searches for the first occurrence of **str** in our string, starting from the **pos** position.

```
int find_c(String* this, char const *str, size_t pos);
```

Searches for the first occurrence of **str** in our string, starting from the **pos** position.

Returns the position at which the occurrence of **str** has been found. Returns -1 if the string has not been found. If the string is too long, it returns -1. If the position is invalid, it returns -1.

Exercise 11

Insert

repository subdir: /ex11
compilation: gcc -Wall -Wextra -Werror -std=gnu99
files to turn in: String.h, String.c
forbidden functions: none
points: 1

Add the following two **insert** member functions. These functions enlarge the string's size. If **pos** is greater than the size of the string, then you must perform an insertion at the end of the string.

```
void insert_c(String* this, size_t pos, char const *str);
```

Copies the content of **str** into the **String** at the **pos** position.

```
void insert_s(String* this, size_t pos, String const *str);
```

Copies the content of **str** into the **String** at the **pos** position.



Watch out for the `'\0'`



Exercise 12

to_int

repository subdir: /ex12
compilation: gcc -Wall -Wextra -Werror -std=gnu99
files to turn in: String.h, String.c
forbidden functions: none
points: 1

Add a **to_int** member function:

```
int to_int(String* this);
```

Transforms the **String** into an **int**. This function will have the same behavior as the **atoi(3)** function.

Exercise 13

Split

repository subdir: /ex13
compilation: gcc -Wall -Wextra -Werror -std=gnu99
files to turn in: String.h, String.c
forbidden functions: none
points: 2

Add the following two **split** member functions.

```
String *split_s(String* this, char separator);
```

Returns a table of strings that correspond to the string that was split by the '**separator**' delimiter.

```
char **split_c(String* this, char separator);
```

Returns a table of character strings that correspond to the string that was split by the '**separator**' delimiter.



Exercise 14

Aff

repository subdir: /ex14
compilation: gcc -Wall -Wextra -Werror -std=gnu99
files to turn in: String.h, String.c
forbidden functions: none
points: 5

Add a **aff** member function:

```
void aff(String* this);
```

This function displays the content of the **String** on the standard output.



Watch out! Carriage returns were never mentioned!
printf is NOT necessarily a good idea!

**Yes, this function is worth the most points :)

Exercise 15

Join

repository subdir: /ex15
compilation: gcc -Wall -Wextra -Werror -std=gnu99
files to turn in: String.h, String.c
forbidden functions: none
points: 2

Add the following two **join** member functions.

```
void join_c(String* this, char delim, char const** tab);
```

This member function will assign a string of characters, composed of all of **tab**'s character strings separated by the **delim** delimiter, to the **String**. The table will always be NULL terminated.

```
void join_s(String* this, char delim, String* tab);
```

This member function will assign a string of characters, composed of all of **tab**'s character strings separated by the **delim** delimiter, to the **String**. The table will always be terminated by an empty **String**.



Exercise 16

Substr

repository subdir: /ex16

compilation: gcc -Wall -Wextra -Werror -std=gnu99

files to turn in: String.h, String.c

forbidden functions: none

points: 3

Add a **substr** member function:

```
String* substr(String *this, int offset, int length);
```

Extracts a sub string, starting from **offset**, with the **length** size.

The function returns the found sub string as a new **String** instance.

If **offset** is negative, it represents the number of characters starting from the end.

If **length** is negative, it represents the number of characters to be copied from the left of the **offset**.

If the specified sub string is partially outside the **String**, you just return the part that is in the **String**.