Problem A
Crazy Tree

The aim of the system is to read, decode (character by character) and store every encoded text file in Crazy Tree, an ordered data structure in form a tree constituted by lists. The input file (ASCII - *American Standard Code for Information Interchange*) consists of printable characters between 33 and 255, defined as an integer. Conversely, it is only considered characters between 32 and 122. In the ASCII table, 32 characters represents the 'space', while the range from 33(!) to 122(z) comprises the entire alphabet without accents.

The decoded text is stored in our Crazy Tree, whose nodes have 0, 1, 2 or 3 children. Each tree node stores one word and isolates spaces (032) at the final range. Therefore, should we only have one word, just from the root is Crazy Tree built. Nevertheless, the tree is insane, therefore the node cannot store the entire word just by itself. In order to store correctly, it is used a list, in which each list node stores one character of the written word. Lastly, were there to be any word censored, the application removes it entirely. Besides, the decoded text is printed.

Crazy Tree representation

Represented by the Figure 1, the crazy tree is input file ABC [1], while the crazy.in consists of the Figure 2.

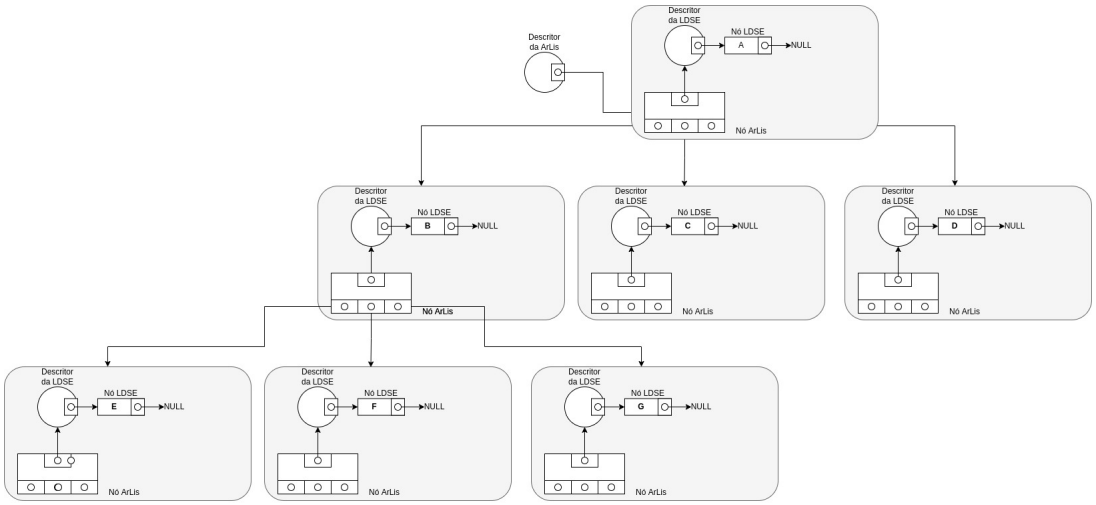

Figure 1. Crazy Tree results to the input file (*ABC.in*).

Tools

In the system packet, it is provided tools to encode input files (input-generator) and create new (text-file-creator).

[1] File *ABC.in* located in examples directory.

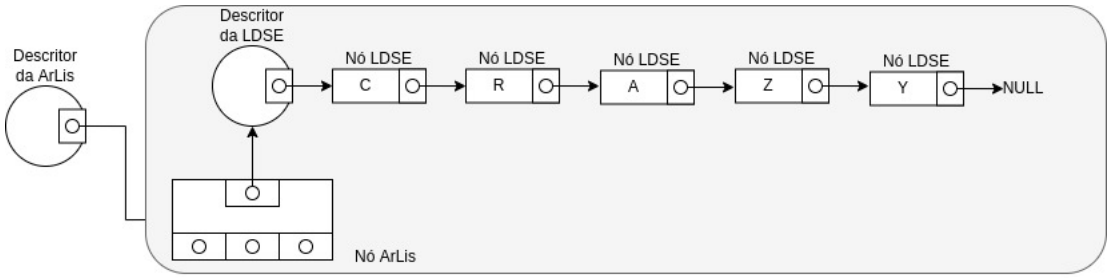

Figure 2. Crazy Tree results to the input file (*crazy.in*).

Creating a new text file

In order to use it, you need to review the settings file (*tools/words.map*) and define the number of characters. This tool writes random characters (ASCII between 32 and 122) and the words described in *tools/words.map* on the output file (newfile).

```
$ make clean
$ make
$ tools/text-file-creator 10000
$ The generated file is newfile.
```

Generating input file to sysCT

This application reads a text file, encodes character by character and writes in other new file. The output file has the same name with a new extension *.in*. To use it, you need to follow the sequencing steps:

```
$ make clean
$ make
$ tools/input-generator examples/ABC
$ The encoded file(examples/ABC.in) was successfully generated!!
```

In this case, we choose the ABC file.

Setting phase

In this phase, you must verify the input files.

```
$ cat tools/words.map
sun
car
writer
motivation
Strawberry
Friendship
Everything
Appreciate
Palatopharyngoplasty
incontrovertiblenes
```

```
$ cat examples/ABC
A B C D E F G H I J K L M N O P Q R S T U V Y X Z

$ cat examples/ABC.in
0650320660320670320680320690320700320710320720320730320740
3207503207603207703207803207903208003208103208203208303208
4032085032086032089032088032090010
$ sysCT < examples/ABC.in
```

Execute phase

```
$ make clean
$ make
$ sysCT < examples/ABC.in
```

Output phase

The system output shows the removed words (described in tools/words.map) or ERROR (not found words). Finally, the new text without the removed words is printed.

```
$ sysCT < examples/ABC.in
$Removing :(ERROR - sun)
(ERROR - car)
(ERROR - writer)
(ERROR - motivation)
(ERROR - Strawberry)
(ERROR - Friendship)
(ERROR - Everything)
(ERROR - Appreciate)
(ERROR - Palatopharyngoplasty)
(ERROR - incontrovertiblenes)
**************
A B C D E F G H I J K L M N O P Q R S T U V Y X Z
**************
```