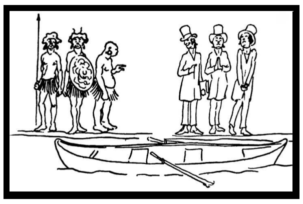
`MSc (Computing Science) 2013-2014`
`C/C++ Laboratory Examination`

Imperial College London

Monday 13 January 2014, 15h00 – 17h10



☞ You are given 10 minutes reading time during which you may read through the questions.

☞ You must complete and submit a working program by 17h10.

☞ Log into the Lexis exam system using your DoC login as both your login and as your password (**do not use your usual password**).

☞ You are required to add to the pre-supplied header file **river.h**, pre-supplied implementation file **river.cpp** and to create a **makefile** according to the specifications overleaf.

☞ You will find the source files **river.cpp**, **river.h** and **main.cpp**, and the data files **bank.txt**, **boat.txt**, **cannibal.txt**, **missionary.txt**, **pot.txt**, **river.txt** and **sun.txt** in your Lexis home directory (**/exam**). If one of these files is missing alert the invigilators.

☞ **Save your work regularly.**

☞ Please log out once the exam has finished. No further action needs to be taken to submit your files.

☞ No communication with any other student or with any other computer is permitted.

☞ You are not allowed to leave the lab during the first 30 minutes or the last 30 minutes.

☞ **This question paper consists of 7 pages.**

Image Credit: `http://www.retronaut.com/2012/11/three-cannibals-and-three-missionaries-arrive-at-a-riverhow-do-they-get-across/`

Problem Description

Figure 1: The missionaries and cannibals river-crossing puzzle has led to many fun online games like this one. Relax after the examination by playing it at `http://www.plastelina.net/`.

Your challenge is to code a classical river-crossing puzzle dating from the end of the 19th century. As shown in Figure 1, the puzzle concerns three missionaries and three cannibals who must cross a river from the left bank to the right bank using a boat which carries at most two people. The boat cannot cross the river by itself with noone on board. Swimming is impossible since the river contains piranhas and sharks.

Importantly, for both banks, if there are missionaries present on a bank, then there must **not** be more cannibals than missionaries on that bank (since then the cannibals eat the missionaries[1]).

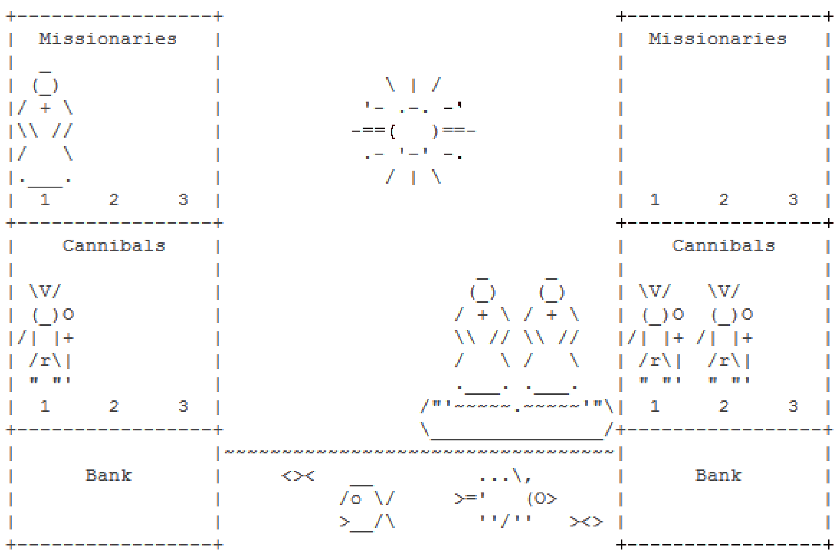


Figure 2: Missionaries and cannibals in ASCII art.

To enliven the presentation, Figure 2 shows how we are going to make use of ASCII-art scenes (actually 2D arrays of characters) to represent the puzzle.

[1]Remarkably, as it turns out, this is **not** a purely hypothetical scenario; later you may wish to look up e.g. `http://www.telegraph.co.uk/news/worldnews/1560483/Cannibal-tribe-apologises-for-eating-Methodists.html`

Pre-supplied functions and files

To get you started, you are supplied with some ASCII-art drawings in the files **bank.txt** (river bank), **boat.txt** (boat used to transport missionaries and cannibals across the river), **cannibal.txt** (a cannibal), **missionary.txt** (a missionary), **pot.txt** (a pot, used to demonstrate the scene creation functions), **river.txt** (the river), and **sun.txt** (the sun). You can use the UNIX command **cat** to inspect these files, e.g.

```
% cat missionary.txt
  _
 (_)
/ + \
\\ //
/   \
.___.
```

You are also supplied with some helper functions (with prototypes in **river.h** and implementations in the file **river.cpp**):

- `char **create_scene()` is a function which allocates memory for and initialises an empty ASCII-art scene. The return value is a 2D array of characters with `SCENE_HEIGHT` rows and `SCENE_WIDTH` columns (constants defined in `river.h`).
- `destroy_scene(char **scene)` frees the memory allocated by a call to `create_scene()`.
- `void print_scene(char **scene)` displays the ASCII-art scene stored in the 2D array of characters `scene`.
- `bool add_to_scene(char **scene, int row, int col, const char *filename)` inserts the ASCII-art drawing stored in the file named `filename` into the given `scene` starting at row–column coordinates (`row`,`col`).
- `const char *status_description(int code)` provides human readable strings describing the status code passed as a parameter. The status codes themselves are declared in **river.h**, and will be referred to in Questions 2 and 3.

You are also supplied with a main program in **main.cpp**.

Specific Tasks

1. Write a function `make_river_scene(left, boat)` which takes two character strings: `left` describing the contents of the *left* river bank, and `boat` describing the contents of the boat[2], and which returns a corresponding ASCII-art scene. The character string for the left bank may be up to 7 letters long and may contain three different kinds of characters: `M` denoting the presence of a missionary, `C` denoting the presence of a cannibal and `B` denoting that the boat is at the left bank. The character string for the boat may be up to 2 letters long and may contain only the characters `M` and `C`. Note both strings may also be empty and the order of the characters is irrelevant. **Hint:** Start by drawing the left river bank at (0,0), the right river bank at (0,53) the sun at (3,30), and the river at (19,19)[3].

   For example, the code:

   ```
   scene = make_river_scene("CM","MM");
   print_scene(scene);
   ```

   should display output similar to that shown in Figure 2. The initial state of the puzzle (with all missionaries and cannibals and the boat at the left bank) has `left_bank` set to `"BCCCMMM"` and `boat` set to `""` while the goal state of the puzzle has `left_bank` set to `""` and `boat` set to `""`.

2. Write a function `perform_crossing(left, targets)` which performs a crossing of the river by one or two target entities[4] from one bank of the river to the other using the boat. The parameter `left` is both an input and an output parameter: it initially describes the contents of the *left* river bank, but is modified to reflect the result of the move (including an update on the boat's position). The parameter `targets` is a string with one or two characters drawn from the letters `M` and `C` according to the entities performing the crossing (N.B. performing a "crossing" without anyone on the boat is not allowed).

[2] Note that since there are three missionaries and three cannibals, the contents of the right river bank can be inferred from whatever is not on the left bank or on the boat.

[3] Do not be alarmed if the fish in the river appear to move spontaneously – they like to swim about!

[4] By "entity" we mean a missionary or a cannibal.

```
Loading the boat...
+-----------------+                                  +-----------------+
|  Missionaries   |                                  |  Missionaries   |
|  _              |                                  |                 |
| (_)             |              \ | /               |                 |
|/ + \            |            '- .-. -'             |                 |
|\\ //            |           -==(   )==-            |                 |
|/   \            |            .- '-' -.             |                 |
|.___.            |              / | \               |                 |
|  1     2     3  |                                  |  1     2     3  |
+-----------------+                                  +-----------------+
|    Cannibals    |                                  |    Cannibals    |
|                 |     _     _                      |                 |
|                 |    (_)   (_)                     | \V/   \V/   \V/ |
|                 |   / + \ / + \                    | (_)O  (_)O  (_)O|
|                 |   \\ // \\ //                    |/| |+ /| |+ /| |+|
|                 |   /   \ /   \                    | /r\|  /r\|  /r\||
|                 |   .___. .___.                    | " "'  " "'  " "'|
|  1     2     3  |/"'~~~~~.~~~~~'"\                 |  1     2     3  |
+-----------------+\_______________/                 +-----------------+
|                 |~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~|                 |
|      Bank       |     <><   __         ...\,       |      Bank       |
|                 |          /o \/     >='   (O>     |                 |
|                 |          >__/\       ''/''   ><> |                 |
+-----------------+                                  +-----------------+

Crossing the river...
+-----------------+                                  +-----------------+
|  Missionaries   |                                  |  Missionaries   |
|  _              |                                  |                 |
| (_)             |              \ | /               |                 |
|/ + \            |            '- .-. -'             |                 |
|\\ //            |           -==(   )==-            |                 |
|/   \            |            .- '-' -.             |                 |
|.___.            |              / | \               |                 |
|  1     2     3  |                                  |  1     2     3  |
+-----------------+                                  +-----------------+
|    Cannibals    |                                  |    Cannibals    |
|                 |                      _     _     |                 |
|                 |                     (_)   (_)    | \V/   \V/   \V/ |
|                 |                    / + \ / + \   | (_)O  (_)O  (_)O|
|                 |                    \\ // \\ //   |/| |+ /| |+ /| |+|
|                 |                    /   \ /   \   | /r\|  /r\|  /r\||
|                 |                    .___. .___.   | " "'  " "'  " "'|
|  1     2     3  |                 /"'~~~~~.~~~~~'"\|  1     2     3  |
+-----------------+                 \_______________/+-----------------+
|                 |~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~|                 |
|      Bank       |     __     ><>            ...\,  |      Bank       |
|                 |    /o \/               >='   (O> |                 |
|                 |    >__/\         <><      ''/''  |                 |
+-----------------+                                  +-----------------+

Unloading the boat...
+-----------------+                                  +-----------------+
|  Missionaries   |                                  |  Missionaries   |
|  _              |                                  |  _     _        |
| (_)             |              \ | /               | (_)   (_)       |
|/ + \            |            '- .-. -'             |/ + \ / + \      |
|\\ //            |           -==(   )==-            |\\ // \\ //      |
|/   \            |            .- '-' -.             |/   \ /   \      |
|.___.            |              / | \               |.___. .___.      |
|  1     2     3  |                                  |  1     2     3  |
+-----------------+                                  +-----------------+
|    Cannibals    |                                  |    Cannibals    |
|                 |                                  |                 |
|                 |                                  | \V/   \V/   \V/ |
|                 |                                  | (_)O  (_)O  (_)O|
|                 |                                  |/| |+ /| |+ /| |+|
|                 |                                  | /r\|  /r\|  /r\||
|                 |                                  | " "'  " "'  " "'|
|  1     2     3  |                 /"'~~~~~.~~~~~'"\|  1     2     3  |
+-----------------+                 \_______________/+-----------------+
|                 |~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~|                 |
|      Bank       |      __         ><>     ,/...    |      Bank       |
|                 |    \/ o\               <O)   `=< |                 |
|                 |    /\__<    <><         ``\``    |                 |
+-----------------+                                  +-----------------+
The missionaries have been eaten! Oh dear!
```

Figure 3: The three phases of a river crossing. Here two unfortunate missionaries cross the river only to be eaten by the three waiting cannibals upon arrival.

The function assumes an initially empty boat, and performs the move in three phases, each of which should be displayed on the console as an ASCII-art scene:

(a) Loading the boat: The `targets` are transferred from the bank where the boat is, onto the boat.
(b) Crossing the river: The boat crosses the river to the other side.
(c) Unloading the boat: The `targets` are transferred from the boat onto the bank where the boat is now.

The return value should be a status code reflecting the outcome of the crossing operation e.g. `ERROR_INVALID_MOVE` in the case of an invalid `targets` string, `ERROR_MISSIONARIES_EATEN` in the case of the unfortunate missionaries being eaten by cannibals, or `VALID_NON_GOAL_STATE` in the case of a valid but non-goal state.

For example, the code:

```
char left[10] = "MMMB";
int result = perform_crossing(left, "MM");
cout << status_description(result) << endl;
```

should result in the output shown in Figure 3, with `left` set to `"M"` and `result` set to `ERROR_MISSIONARIES_EATEN` (since the two missionaries arriving from the left bank will be outnumbered by the cannibals on the right bank).

3. Write a function `play_game()` which allows a user to play the game of missionaries and cannibals by suggesting boat crossings via the keyboard. The return value of the function should be an appropriate status code e.g. `ERROR_MISSIONARIES_EATEN` or `VALID_GOAL_STATE` (i.e. everyone crossed successfully).

(The three parts carry, respectively, 35%, 35% and 30% of the marks)

What to hand in

Place your function implementations in the file **river.cpp** and corresponding function declarations in the file **river.h**. Use the file **main.cpp** to test your functions. Create a **makefile** which compiles your submission into an executable file called **river**.

Hints

1. You will save time if you begin by studying the main program in **main.cpp**, the header file **river.h**, the pre-supplied functions in **river.cpp** and the data files **bank.txt**, **boat.txt**, **cannibal.txt**, **missionary.txt**, **pot.txt**, **river.txt** and **sun.txt**.
2. All the questions will be **much** easier if you exploit the pre-supplied helper functions.
3. Feel free to define any of your own helper functions which would help to make your code more elegant.
4. Try to attempt all questions. If you cannot get one of the questions to work, try the next one.
5. You are not explicitly required to use recursion in your answers to any of the questions. Of course, however, you are free to make use of recursion if you wish (esp. where it increases the elegance of your solution).

Bonus Challenge (optional)

For extra credit, write a function `find_solution(left,answer)` which outputs in `answer` the comma-separated series of crossings required to complete the game starting from the configuration with left bank contents given by `left` and an empty boat.

The return value should be `VALID_GOAL_STATE` if a solution exists or `ERROR_BONUS_NO_SOLUTION` if a solution does not exist.

This should work with code:

```
char left[10], answer[512];
strcpy(left, "MMMCCCB");
if (find_solution(left, answer)==VALID_GOAL_STATE)
  cout << "Solution is " << answer << endl;
else
 cout << "Solution does not exist" << endl;
```