

Lab 4. Printing Shapes

```
.:ir7v777vr7i:
iLJYYvv7v777L7YLjvi
:UjY77r7r7r7r7r7r7YJu:
i1Lv7v7v7v7v777v7v7LL2r
i5LLLLvYLLLLvYLLvLvL7LvLvFi
.SYYLLLYLYLJYjYjYjYjLYLLYS.
72jLYLJLY7L7vr7r7rvvLLYLYLj17
uFYyLLr7r77YYUju3J77r7rLvYYFU
12Y7vLjJ22qXqSxkqPP2UjUYL7LuS
uuuUFxJU8Z:::XB0JkkFYyU
u5027:.UBB:::BBX.:7jEFF :. .i.
NBj:,:BN.:i:i::2Bv.:788 7U,,7U;i
kB:.,. . .,:i: . . .:BB :v:::L7
jB,.,.,.,.:i: . . .:..00LuYLr;LYJ:
qq:iii;i;iuMBBqi;ir;rii:UPjjUXPU7.
EBR77L77777iY2ui777rv7L777kUU5BBq
YBB25jJvYLYvY77rLLLLJLYLjuqkPMBS:
UBBL5kSu2UUUUuU2u5UF1550NBmj.
UBB808888BMMOMGOOZqZMM7
.YSEMOqOqJ:2XPkNFSF8Er
:L5UjuP1jYJ,,iuvYLJjqU
7i:7uXZ1LY7LvLvLYY2U
:::,iqBBUY7v7v777v7Y1j
iULuN5 EO1LLv7v777LL1U
,50k: MZJY7L7v7777Lu2
. :B1J7777r7r77v7U1
USYrrrrr7rrr7;77F,
FGNZEM8MMMGO80NEr
BB800EO8MBBZ888BBi
:5UJUUXYPX22UkN
XUuu22UYBqF25EP
XGXXSqq:EBMMBB.
. . . . . 5kULjjPi.XBBP. . . . .
. . . . .,,:jJ7L7YjSrrLLiii::: . . . . .
. . . . .,:ir80MMBBBG27riiii::: . . . . .
. . . . .:ir;r::: . . . . .
```

Task A. Box

Write a program `box.cpp` that asks the user to input `width` and `height` and prints a solid rectangular box of the requested size using asterisks.

Also, print a line `Shape:` between user input and the printed shape (to separate input from output).

Example:

```
Input width: 7
Input height: 4
```

```
Shape:
*****
*****
*****
*****
```

Hint:

- First find how to print one row of stars (print the asterisk character `width` times followed by end-of-line).
- Then, once you know how to print one line of stars, repeat it `height` times (using a loop).

Task B. Checkerboard

Write a program `checkerboard.cpp` that asks the user to input `width` and `height` and prints a rectangular checkerboard of the requested size using asterisks and spaces (alternating).

Example:

```
Input width: 11
Input height: 6
```

```
Shape:
* * * * *
  * * * *
* * * * *
  * * * *
* * * * *
  * * * *
```

Hint:

You used nested loops in the previous task that looked probably like

```
for (int row = 0; row < height; row++) {
    for (int col = 0; col < width; col++) {

        ...

    }
}
```

Inside the loops, you can add an `if` statement that will be conditionally printing asterisk `*` or (space) depending on the coordinates `row` and `col`.

Task C. Cross

Write a program `cross.cpp` that asks the user to input the shape `size`, and prints a diagonal cross of that dimension.

Example:

```
Input size: 8
```

```
Shape:
*       *
 *     *
  *   *
   **
  **
 *  *
*   *
*     *
```

Task D. Lower triangle

Write a program `lower.cpp` that prints the bottom-left half of a square, given the side `length`.

Example:

Input side length: 6

Shape:

```
*
**
***
****
*****
*****
```

Task E. Upper triangle

Write a program `upper.cpp` that prints the top-right half of a square, given the side length.

Example:

Input side length: 5

Shape:

```
*****
 ****
  ***
   **
    *

```

Task F. Upside-down trapezoid or triangle

Write a program `trapezoid.cpp` that prints an upside-down trapezoid of given width and height.

However, if the input height is impossibly large for the given width, then the program should report, Impossible shape!

Example 1:

Input width: 12

Input height: 5

Shape:

```
*****
 *****
  *****
   *****
    *****

```

Example 2:

Input width: 5

Input height: 3

Shape:

```
*****
 *****

```

★

Example 2:

```
Input width: 12
Input height: 7
```

Impossible shape!

Hint:

You can start with the number of

```
spaces = 0;
stars = width;
```

On each line, print **that number of spaces** followed by **that number of stars**. After that, the number of spaces gets incremented by 1, while the number of stars gets decremented by 2:

```
spaces += 1;
stars -= 2;
```

Task G. Checkerboard (3x3)

Write a program `checkerboard3x3.cpp` that asks the user to input `width` and `height` and prints a checkerboard of 3-by-3 squares. (It should work even if the input dimensions are not a multiple of three.)

Example 1:

```
Input width: 16
Input height: 11
```

Shape:

```
***      ***      ***
***      ***      ***
***      ***      ***
      ***      ***      *
      ***      ***      *
      ***      ***      *
***      ***      ***
***      ***      ***
***      ***      ***
      ***      ***      *
      ***      ***      *
```

Example 2:

```
Input width: 27
Input height: 27
```

Shape:

```
***      ***      ***      ***      ***
***      ***      ***      ***      ***
***      ***      ***      ***      ***
      ***      ***      ***      ***
```

```

    ***      ***      ***      ***
    ***      ***      ***      ***
***      ***      ***      ***      ***
***      ***      ***      ***      ***
***      ***      ***      ***      ***
    ***      ***      ***      ***
    ***      ***      ***      ***
    ***      ***      ***      ***
***      ***      ***      ***      ***
***      ***      ***      ***      ***
***      ***      ***      ***      ***
    ***      ***      ***      ***
    ***      ***      ***      ***
    ***      ***      ***      ***
***      ***      ***      ***      ***
***      ***      ***      ***      ***
***      ***      ***      ***      ***
    ***      ***      ***      ***
    ***      ***      ***      ***
    ***      ***      ***      ***
***      ***      ***      ***      ***
***      ***      ***      ***      ***
***      ***      ***      ***      ***

```

How to submit your programs.

Each program should be submitted through Gradescope.

Write separate programs for each part of the assignment.

Submit only the source code (.cpp) files, not the compiled executables.

Each program should start with a comment that contains your name and a short program description, for example:

```

/*
Author: your name
Course: CSCI-136
Instructor: their name
Assignment: title, e.g., Lab1A

```

Here, briefly, at least in one or a few sentences describe what the program does.

```

*/

```