

Christopher Figueroa

7/10/23

## HW 5: Creative ASCII Art

### **Overview:**

For this assignment I made utilized 2 prompts, one in ChatGPT and one in Rix, to create 2 separate C++ programs. The purpose of these prompts is to build a program that generates ASCII art that varies during each compilation of the code. Unlike in previous assignments, the final output did not have to match one particular set of requirements. Instead, the goal was to create my own unique randomly generated ASCII art that follows a theme of my choice. As a consequence, the prompts that I have the AIs did not have to be as complex.

The first prompt, the ChatGPT prompt uses the usual “senior software engineer” approach. I include Homework 5’s description of ASCII art in the initial prompt to give the AI more detailed instructions on the program’s requirements. Usually ChatGPT tends to yield the most functional code. However in this instance, the AI had trouble coming up with an effective and efficient way to follow the assignment’s requirements.

The second prompt, the Rix AI prompt contained in “RixPromptHW5.txt”, yielded a successful program despite the AI’s tendency to occasionally reply with error prone code. Rix’s piece by piece approach to populating the matrix proved to be more effective than ChatGPT’s built it all at once approach. The final working code is contained in the C++ file “RixCodeHW5.cpp”.

### **First ChatGPT Prompt:**

For the first ChatGPT prompt, the one contained in “ChatGPTpromptHW5.txt” I prompted the AI to generate a C++ program that generates ASCII art using the instructions from the assignment description. I allowed the AI the freedom to find its own approach toward building the program. The result of the initial query resulted in an output that prints out a random set of characters that populate a matrix of a set size.

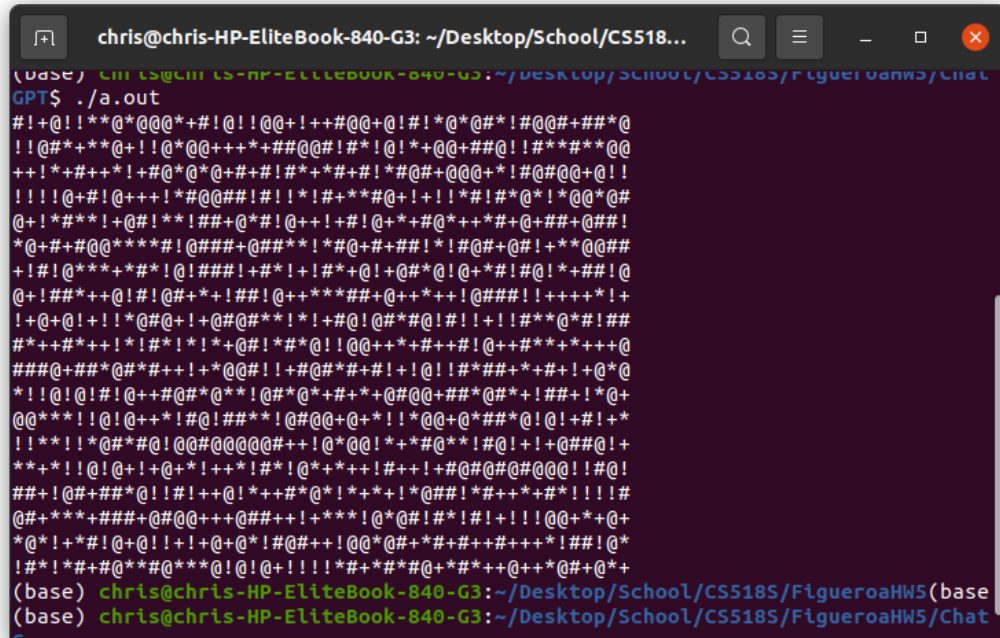
(1). User:

For the following prompt act as a senior software engineer familiar with programming in c++. My goal is create a program that generates ASCII art onto a 2D matrix:

"Develop a C/C++/Python program using your preferred AI(s) that generates stochastic (guided randomization) ASCII art to the terminal, so that each time I run the program, I see a new piece of art with the same theme/style as all the other runs. Specifically, I've become fond of the badly Photoshopped product photos for so many things on Amazon (they're really just collages), so I'm looking for something inspired by that, but ASCII... scenes of a certain type with layered smaller bits of ASCII art that make the whole scene. Placement and shape of things could be made stochastically."

The ASCII art should use a combination of keyboard characters and blank spaces to generate random scenes.

Line 2 ChatGPT Output:



From query lines 3-6, I prompt ChatGPT to alter the characters to add the characters “/ \ [ ] { } \_ -“ to the matrix. I also instruct the AI to randomize the generation of characters throughout the matrix. At this point, I did not yet have a coherent plan on what I wanted the program to generate, so I instructed ChatGPT to generate a circle.

(7). User:

The code generates a new design successfully. However the design that the program generates appears to be disorderly collection of characters, not a coherent art design. How could I modify the code to generate a circle that consists of characters such as “/ \ \_ / ” characters? Or any other geometric shape?

(9). User:

ChatCodeHW5.cpp:7:32: warning: unknown escape sequence: ‘\\_’

```
7 | const std::string asciiChars = "\_ / ";  
  |
```

The warning above comes from the placement of the ‘\\_’ characters. However, this warning does not affect the compilation of the program, it was merely a minor annoyance. To get rid of this minor annoyance, I prompted ChatGPT to find an alternate syntax that eliminates that warning. The screenshot below represents my first attempt at getting the AI to generate a matrix with a defined geometric shape. Keep in mind, this generation is static, not randomized.

Line 10 Circle Output:



In prompt line 11, I instruct ChatGPT to generate the image of a pirate in plain text. The line 12 response below generates something that somewhat resembles a pirate. I spend lines 13-20 trying to generate a some randomly generated variants of the pirate. The results that I end up with ultimately yield some less than satisfying outputs.

(12). ChatGPT:

Certainly! Here's an example of ASCII art that resembles a pirate:

plaintext

This ASCII art represents a pirate with a hat, an eye patch (represented by "@"), and crossed swords (represented by "|-|"). You can use this example as a starting point and further modify or enhance it to fit your specific requirements or create your own unique ASCII art pirate.

Line 20 Randomized Pirate:

```
(base) chris@chris-HP-EliteBook-840-G3: ~/Desktop/School/CS518S/FigueroaHW5/ChatGPT$ g++ ChatCodeHW5.cpp
(base) chris@chris-HP-EliteBook-840-G3:~/Desktop/School/CS518S/FigueroaHW5/ChatGPT$ ./a.out
```

```
      _ _
     /-----\
    /          \
   |            | (@
   |            |
   |            |
   |__|||_|_   |
   |  :: /      |
   \ - \        /
    || |_-      /
```

```
(base) chris@chris-HP-EliteBook-840-G3:~/Desktop/School/CS518S/FigueroaHW5$
(base) chris@chris-HP-EliteBook-840-G3:~/Desktop/School/CS518S/FigueroaHW5/ChatGPT$
```

From dialog lines 21-23, I instruct ChatGPT to expand the dimensions of the pirate image. I tell the AI to put a line through the background of the pirate's head which represents the horizon. The result that I end up with resembles the "big head" pirate image from Line 20. In line 24, I send in a query to generate a plain text image of pirate with a parrot with an island background, but that I get is a set of separate images

Line 24 Plaintext:

$$\begin{array}{c} \begin{array}{c} \text{---} \\ | \text{ } \text{ } | \\ | \text{ } \text{ } | \\ // \quad \backslash \backslash \\ (| \quad |) \\ \wedge \quad \wedge \\ \backslash \quad \backslash \end{array} \\ = \\ \begin{array}{c} \text{---} \\ / \quad \backslash \\ | \quad | \\ \backslash \quad / \\ \text{---} \\ | \quad | \\ | \quad | \end{array} \end{array}$$

[illegible]









## Rix AI Prompt:

My second prompt, contained in “RixPromptHW5.txt”, involved the same initial query as the one used in “ChatGPTpromptHW5.txt”. The outcome of this initial query yielded an output that consisted mostly of blank spaces with the exception of one column that consisted of ‘#’ characters.

Line 2 Rix First Code:

```
#include <iostream>
#include <random>

const int WIDTH = 80;
const int HEIGHT = 30;

char board[HEIGHT][WIDTH];

void generateScene() {
    // Fill board with spaces
    for (int i = 0; i < HEIGHT; i++) {
        for (int j = 0; j < WIDTH; j++) {
            board[i][j] = ' ';
        }
    }

    // Generate random ASCII art objects and place on board
    std::random_device rd;
    std::mt19937 gen(rd());

    // Generate random tree
    int treeWidth = std::uniform_int_distribution<int>(5,10)(gen);
    int treeHeight = std::uniform_int_distribution<int>(10, HEIGHT-5)(gen);
    int treeX = std::uniform_int_distribution<int>(0, WIDTH-treeWidth)(gen);
    for (int i = 0; i < treeHeight; i++) {
        board[i][treeX] = '#';
    }

    // Generate random house
    // ...
}

void printScene() {
    for (int i = 0; i < HEIGHT; i++) {
        for (int j = 0; j < WIDTH; j++) {
            std::cout << board[i][j];
```

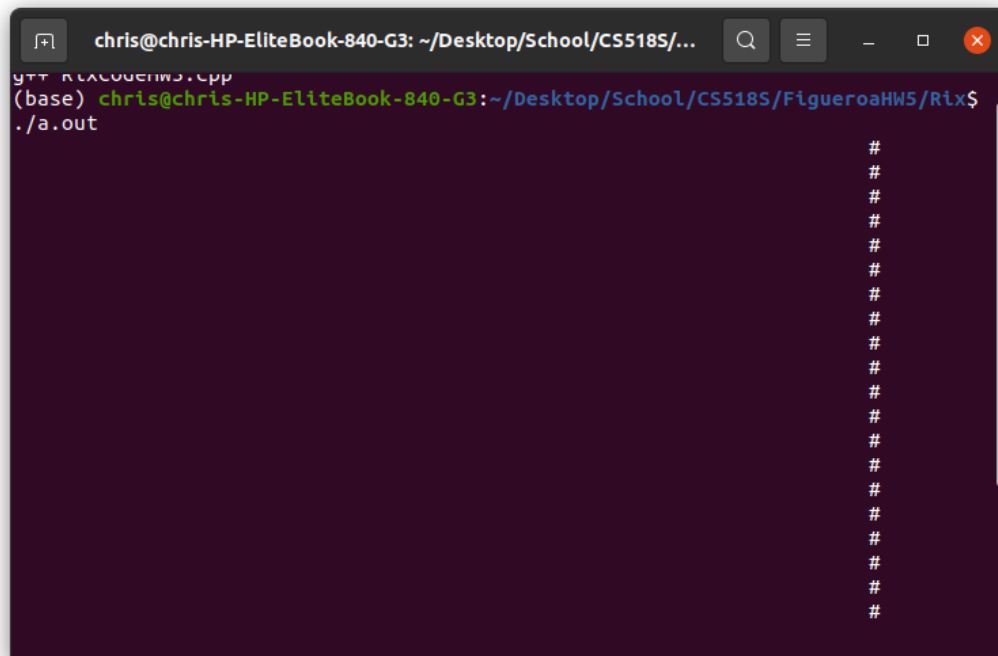
```

    }
    std::cout << std::endl;
}
}

int main() {
    generateScene();
    printScene();
}

```

Line 2 Rix Output:



```

chris@chris-HP-EliteBook-840-G3: ~/Desktop/School/CS518S/...
ytf RixCodeHW5.Cpp
(base) chris@chris-HP-EliteBook-840-G3:~/Desktop/School/CS518S/FigueroaHW5/Rix$
./a.out
#
#
#
#
#
#
#
#
#
#
#
#
#
#
#
#

```

In an effort to obtain a more successful outcome than the one I received from the ChatGPT prompt, I decided to take a piece by piece approach toward the construction of the randomized ASCII art generator program. The lines below represent the initial set of instructions I gave Rix to construct the ocean background, and then the islands and pirates.

(3). User:

Alter the code so it generates the characters "\\_/-" inside a 2D matrix. This 2D matrix should have an image of a pirate with an island in the middle of an ocean in the background.

(5). User:

Is there a way to randomize the position and size of the island, to randomize the number of '~' wave characters, and to randomize the generation of the pirate using a random seed generator?

(7). User:

I noticed that there is a "..." after the // Pirate section of the generateScene function. Is there some code there that needs to be added?

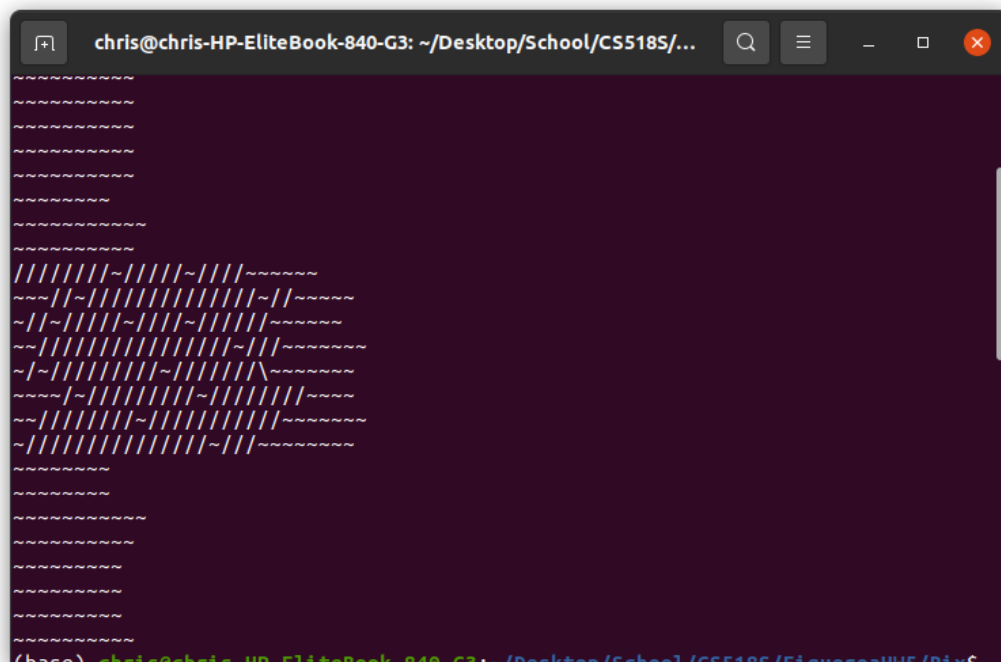
(9). User:

In the Ocean waves section of the code, I am noticing that the waves only appear in the first column of the matrix. Is there a way to fix this issue so the waves appear in more columns?

```
""""  
// Ocean waves  
int numWaves = std::uniform_int_distribution<int>(5,15)(gen);  
for (int i = 0; i < numWaves; i++) {  
    int wavePos = std::uniform_int_distribution<int>(0, WIDTH)(gen);  
    board[i][wavePos] = '~';  
}""""
```

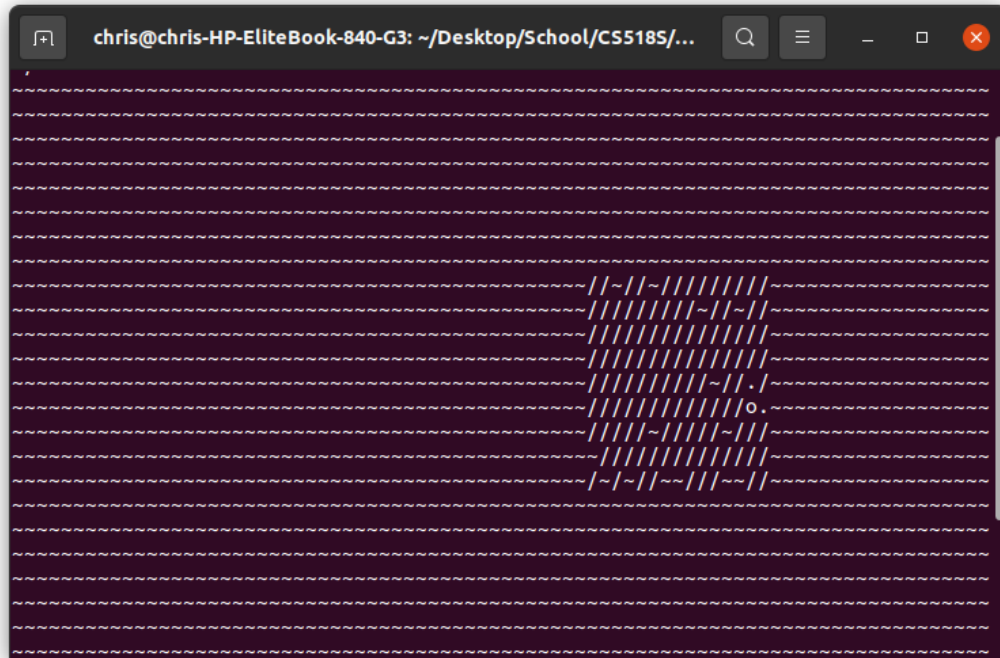
The code that Rix generates at first only creates waves in the first column. The pirate image that the AI generates also only occupies a small segment of the image, usually comprised of a few characters such as “\o”. The line 10 Rix response generates a matrix populated by multiple ‘~’ wave characters. However, these characters do not fill up the entire background.

Line 10 Waves:



I instruct Rix in line 11 to set the default background matrix character to ‘~’. The AI’s response to that request yields a matrix that populates the matrix with the desired ocean background by default. The screenshot below represents the change in output. The generation of the ocean background gives me the idea to generate every piece of the ASCII art segment by segment. I also noticed that Rix has an easier time coming up with code as long as the desired shapes are geometric in nature.

Line 12 Ocean Background:



I made a slight manual alteration in line 13 to allow for the generation of random small spaces in the oceanic background. I also give Rix an instruction to generate 1 to 5 rectangular islands that vary in size during each compilation of the code. The AI in return yields code that outputs the following error:

Line 14 Code:

```
// Island
int numIslands = std::uniform_int_distribution<int>(1,5)(gen);
for (int i = 0; i < numIslands; i++) {
    int islandWidth = std::uniform_int_distribution<int>(10,30)(gen);
    int islandHeight = std::uniform_int_distribution<int>(5,10)(gen);
    int islandX = std::uniform_int_distribution<int>(0, WIDTH - islandWidth)(gen);
    int islandY = std::uniform_int_distribution<int>(5, HEIGHT - islandHeight)(gen);

    for (int j = islandY; j < islandY + islandHeight; j++) {
        for (int k = islandX; k < islandX + islandWidth; k++) {
            board[j][k] = '/';
        }
    }
}
```

Line 15 Error:

RixCodeHW5.cpp: In function 'void generateScene()':  
RixCodeHW5.cpp:46:54: error: 'islandX' was not declared in this scope; did you mean 'srand'?

```

46 |   int pirateX = std::uniform_int_distribution<int>(islandX, islandX + islandWidth)(gen);
    |                                     ^~~~~~
    |                                     srand
RixCodeHW5.cpp:46:73: error: 'islandWidth' was not declared in this scope
46 |   ateX = std::uniform_int_distribution<int>(islandX, islandX + islandWidth)(gen);
    |                                     ^~~~~~

RixCodeHW5.cpp:47:54: error: 'islandY' was not declared in this scope; did you mean 'srand'?
47 |   int pirateY = std::uniform_int_distribution<int>(islandY, islandY + islandHeight)(gen);
    |                                     ^~~~~~
    |                                     srand
RixCodeHW5.cpp:47:73: error: 'islandHeight' was not declared in this scope
47 |   ateY = std::uniform_int_distribution<int>(islandY, islandY + islandHeight)(gen);
    |                                     ^~~~~~

```

I noticed that the placement of the initialization of the int island variables outside of the for loop was causing the error. I informed the AI of the error, and I received only a portion of that code in response. I queried Rix again to correct this issue, however with every new correction, another problem revealed itself. The lines below represent my requests to correct the error.

(17). User:

Show the entire code without the error

(19). User :

I see no difference, why am i still getting not declared in scope error?

(21). User:

Declaring the variables outside the for loop corrected the error. But now the number of islands generated remains at 1. What reason could there be for this? What changes need to be made to the code below?

(23). User:

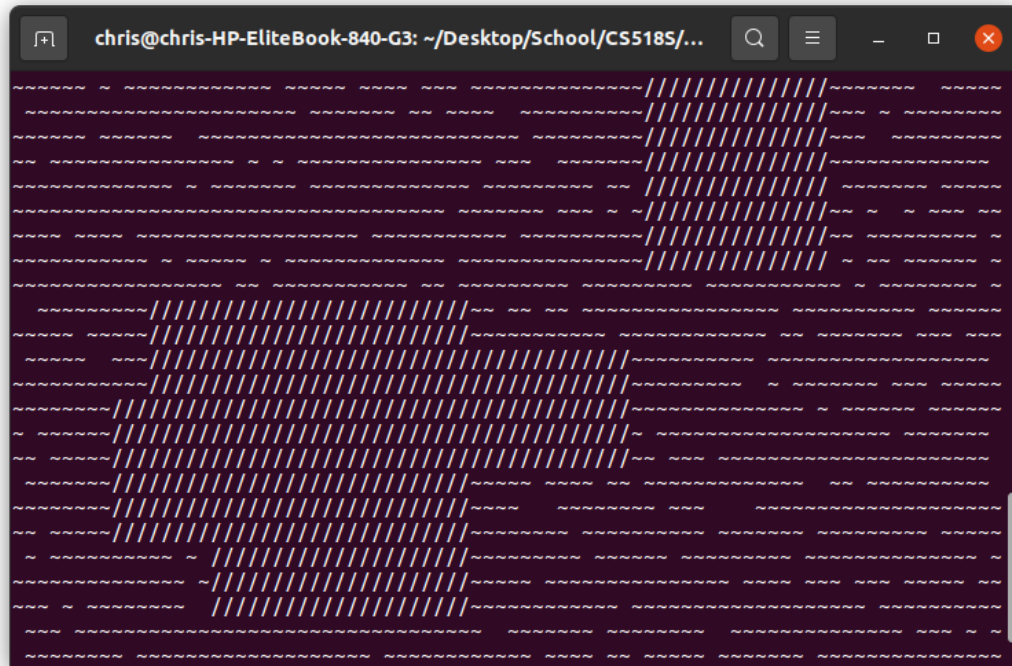
Putting the island variables inside the for loop results in a scope error. Is there another way to do this?

(25). User:

fill up every ... with actual code

The exchange above exemplifies Rix's tendency to provide code with errors. In line 26, I get code that displays an error similar to the one in line 15. When I ask about the cause of the error, Rix says that the declaration of the pirateX and pirateY variables in two locations is causing the issue. To resolve this, I comment out the redundant pirate variables. The removal of the redundant variables resulted in code that generates multiple islands.

Line 29 Islands:



After the correction of that error, the rest of the process of building the ASCII art generation went relatively smoothly. From lines 30-40, I instruct Rix to populate the upper third section of the matrix with blanks before adding in clouds composed of “(\_\_\_\_)” characters. I then instruct the AI to populate the islands with palm trees that stem from the islands themselves. In the last 5 lines I run into some issues with the generation of some upside down palm trees, but Rix corrects those issues with some instructions.

The final code of Rix contains comments // that mark sections that represent the generation of the ocean, sky, islands, clouds, and palm trees. The generation of the “pirates” becomes less relevant since I shift the focus of the generation toward figures with more easily generated geometric shapes.

Line 40 Final Code:

```
#include <iostream>
#include <random>

const int WIDTH = 80;
const int HEIGHT = 30;

char board[HEIGHT][WIDTH];

void generateScene() {
    std::random_device rd;
    std::mt19937 gen(rd());

    // Initialize board with ocean
```

```

for (int i = 0; i < HEIGHT; i++) {
    for (int j = 0; j < WIDTH; j++) {
        board[i][j] = '~'; // Ocean by default
    }
}

// Ocean waves
int numWaves = std::uniform_int_distribution<int>(5,15)(gen);
for (int i = 0; i < numWaves; i++) {
    int wavePos;
    for (int j = 0; j < HEIGHT; j++) {
        wavePos = std::uniform_int_distribution<int>(0, WIDTH)(gen);
        board[j][wavePos] = ' ';
    }
}

// Island
int numIslands = std::uniform_int_distribution<int>(1,5)(gen);

int islandWidth[numIslands];
int islandHeight[numIslands];
int islandX[numIslands];
int islandY[numIslands];

for (int i = 0; i < numIslands; i++) {

    islandWidth[i] = std::uniform_int_distribution<int>(10,30)(gen);
    islandHeight[i] = std::uniform_int_distribution<int>(5,10)(gen);
    islandX[i] = std::uniform_int_distribution<int>(0, WIDTH - islandWidth[i])(gen);
    islandY[i] = std::uniform_int_distribution<int>(5, HEIGHT - islandHeight[i])(gen);

    for (int j = islandY[i]; j < islandY[i] + islandHeight[i]; j++) {
        for (int k = islandX[i]; k < islandX[i] + islandWidth[i]; k++) {
            board[j][k] = ' ';
        }
    }
}

//sky
for (int j = 0; j < HEIGHT/3; j++) {
    for (int k = 0; k < WIDTH; k++) {
        board[j][k] = ' ';
    }
}

int numClouds = std::uniform_int_distribution<int>(3,8)(gen);

```

```

    for (int i = 0; i < numClouds; i++) {
        int j = std::uniform_int_distribution<int>(0, HEIGHT/3-1)(gen);
        int k = std::uniform_int_distribution<int>(0, WIDTH-6)(gen);

        board[j][k] = '(';
        board[j][k + 1] = ' ';
        board[j][k + 2] = ' ';
        board[j][k + 3] = ' ';
        board[j][k + 4] = ' ';
        board[j][k + 5] = ')';
    }

// Palm tree
int treeHeight = 5; // Fixed height for example

int treeX = islandX[i];
int treeY = islandY[i]; // Start from top of island

for (int j = 0; j < treeHeight; j++) {
    treeY--;
    board[treeY][treeX] = '|';
}

board[treeY][treeX + 1] = '!';
board[treeY][treeX + 1] = '/';
board[treeY][treeX + 2] = '\\';
}

// Access coordinates for a random island
int randIsland = std::uniform_int_distribution<int>(0, numIslands-1)(gen);
int pirateX = islandX[randIsland];
int pirateY = islandY[randIsland];

// Pirate
//int pirateX = std::uniform_int_distribution<int>(islandX, islandX + islandWidth)(gen);
//int pirateY = std::uniform_int_distribution<int>(islandY, islandY + islandHeight)(gen);

// Randomly choose one of 4 pirate poses
int pose = std::uniform_int_distribution<int>(0,3)(gen);
if (pose == 0) {
    board[pirateY][pirateX] = '/';
} else if (pose == 1) {
    board[pirateY][pirateX] = '|';
} else if (pose == 2) {
    board[pirateY][pirateX + 1] = '\\';
} else {

```



$$\}$$

Line 40 Final Output:

[illegible]