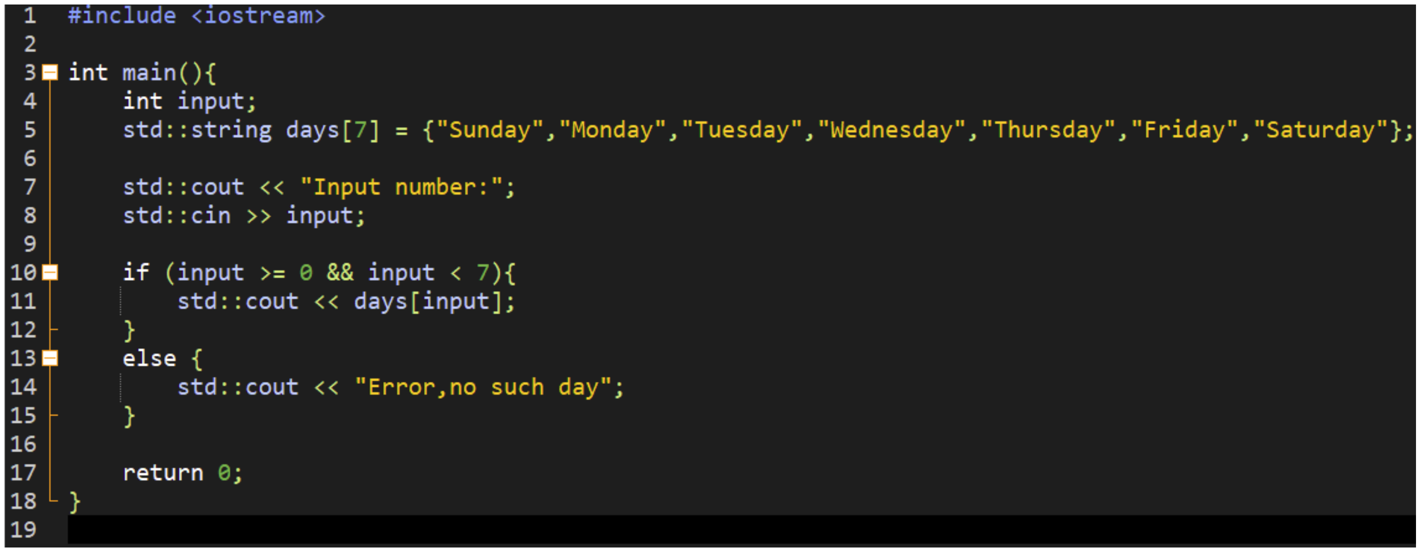
Hands-on Activity 4.3	
Sorting and Searching Arrays	
Course Code: CPE007	**Program:** Computer Engineering
Course Title: Programming Logic and Design	**Date Performed:9/16/25**
Section:CPE11S1	**Date Submitted:9/18/25**
Name(s): Niel Vincent B. Condino	**Instructor: Engr. Jimlord M. Quejado**

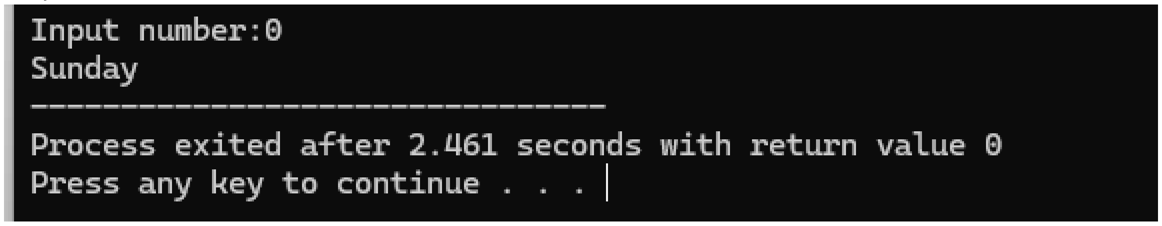
6. Output

1.Code

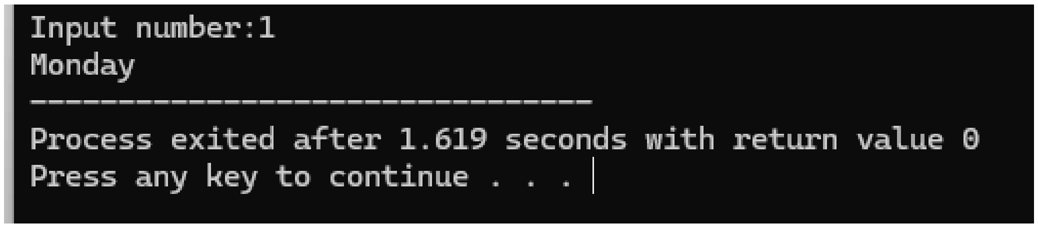


```cpp
#include <iostream>

int main(){
    int input;
    std::string days[7] = {"Sunday","Monday","Tuesday","Wednesday","Thursday","Friday","Saturday"};

    std::cout << "Input number:";
    std::cin >> input;

    if (input >= 0 && input < 7){
        std::cout << days[input];
    }
    else {
        std::cout << "Error,no such day";
    }

    return 0;
}
```

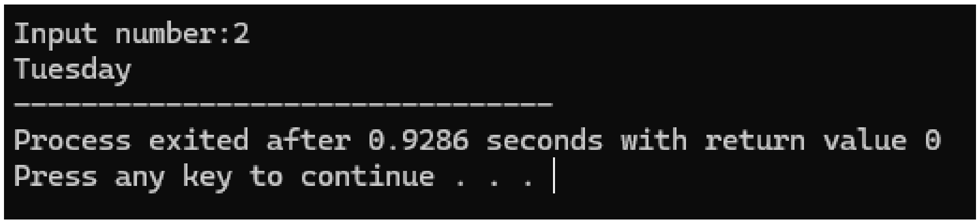
Output



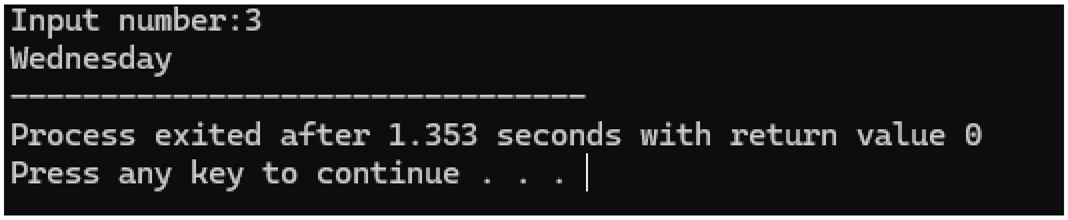
```
Input number:0
Sunday
--------------------------------
Process exited after 2.461 seconds with return value 0
Press any key to continue . . .
```



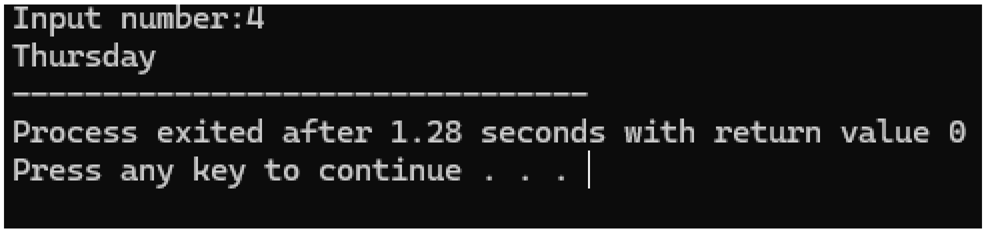
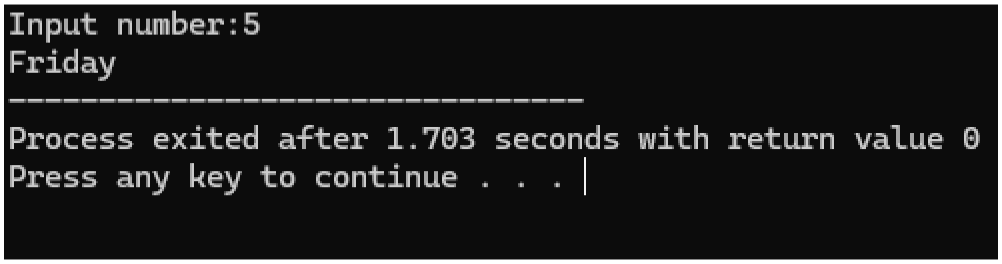
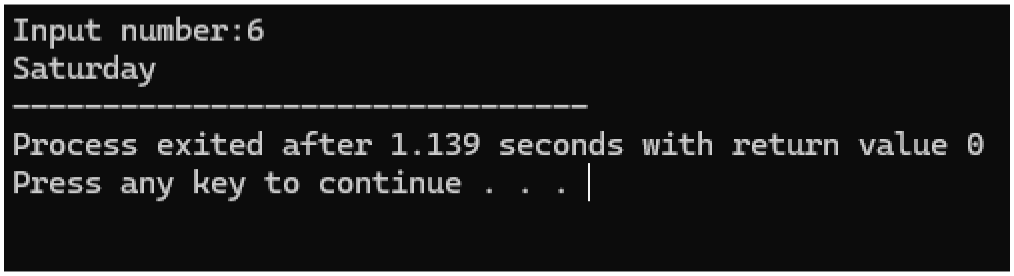
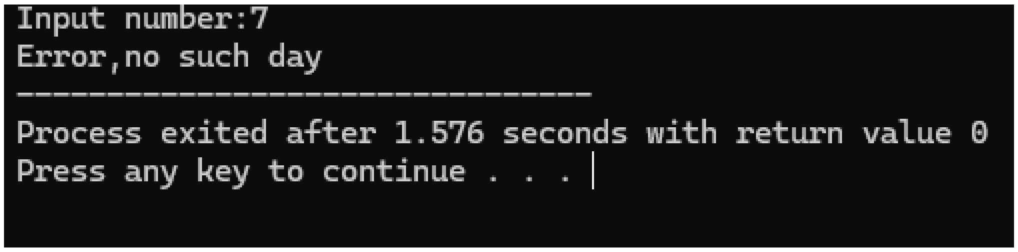
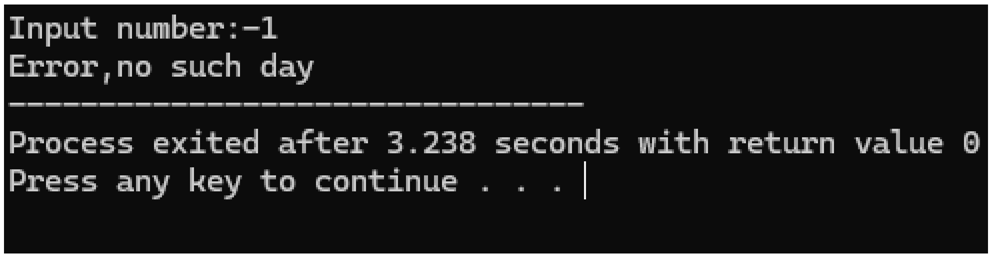
```
Input number:1
Monday
--------------------------------
Process exited after 1.619 seconds with return value 0
Press any key to continue . . .
```



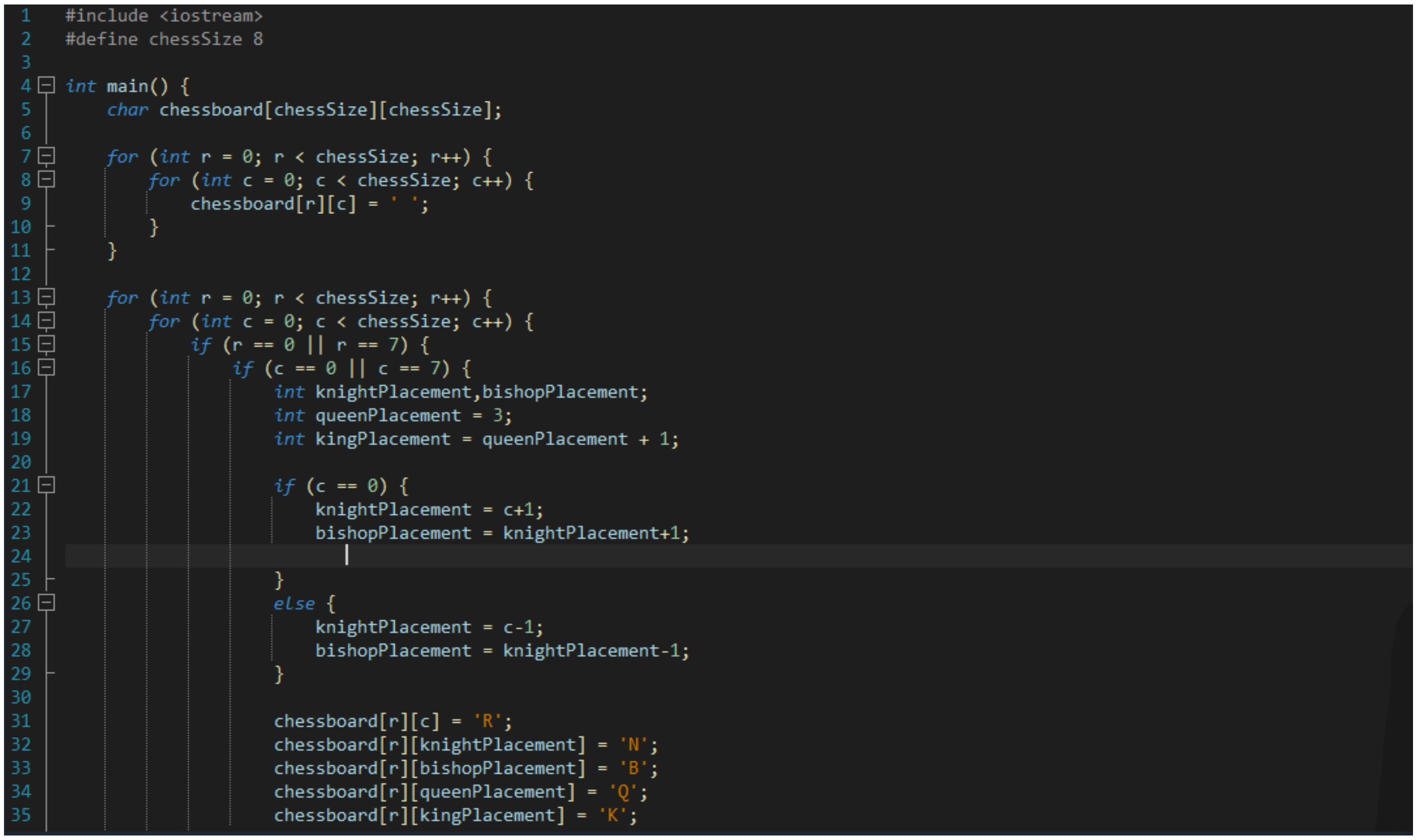
```
Input number:2
Tuesday
--------------------------------
Process exited after 0.9286 seconds with return value 0
Press any key to continue . . .
```



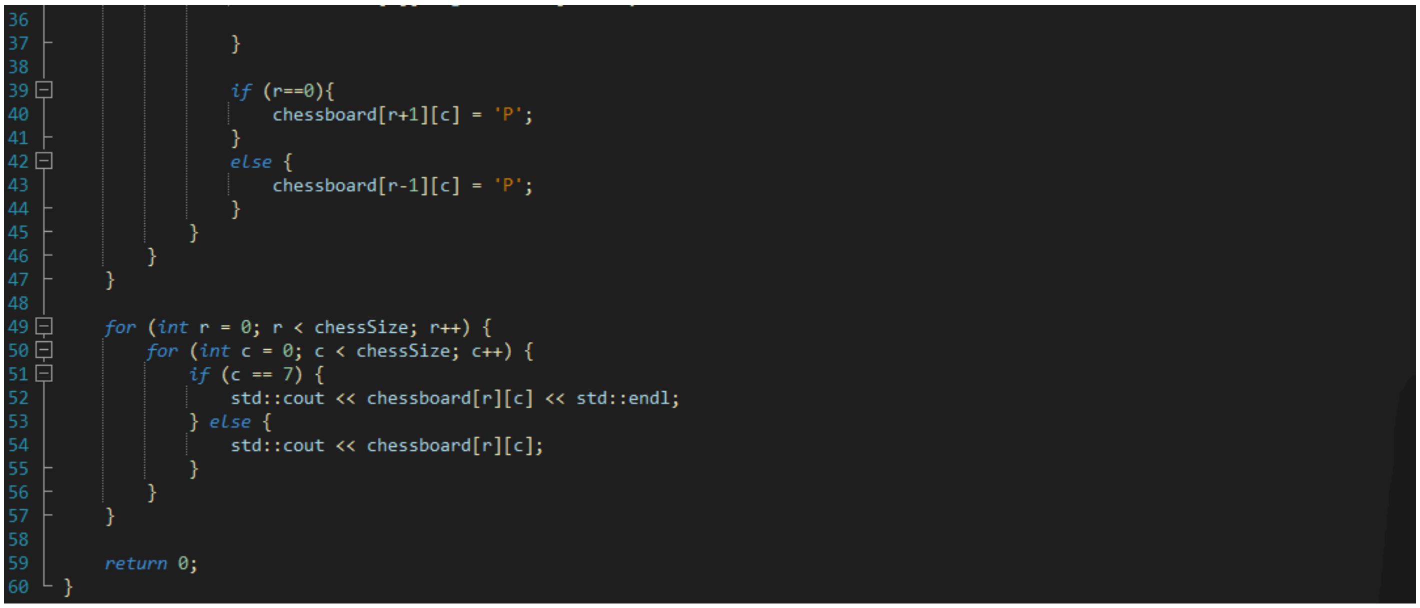
```
Input number:3
Wednesday
--------------------------------
Process exited after 1.353 seconds with return value 0
Press any key to continue . . .
```


```
Input number:4
Thursday
--------------------------------
Process exited after 1.28 seconds with return value 0
Press any key to continue . . .
```

```
Input number:5
Friday
--------------------------------
Process exited after 1.703 seconds with return value 0
Press any key to continue . . .
```

```
Input number:6
Saturday
--------------------------------
Process exited after 1.139 seconds with return value 0
Press any key to continue . . .
```

```
Input number:7
Error,no such day
--------------------------------
Process exited after 1.576 seconds with return value 0
Press any key to continue . . .
```

```
Input number:-1
Error,no such day
--------------------------------
Process exited after 3.238 seconds with return value 0
Press any key to continue . . .
```

2.Code

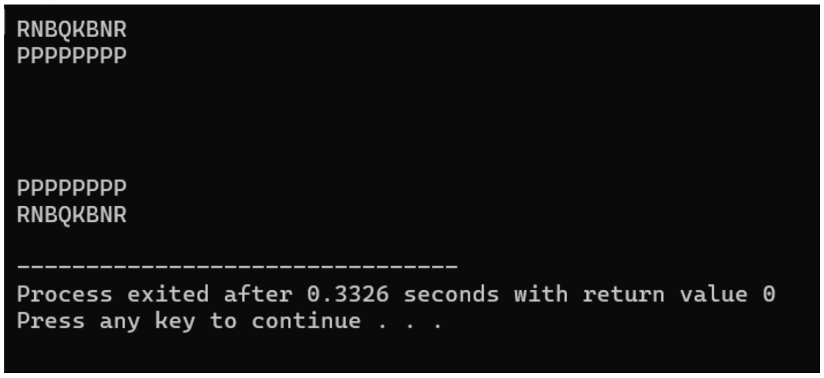


```cpp
#include <iostream>
#define chessSize 8

int main() {
    char chessboard[chessSize][chessSize];

    for (int r = 0; r < chessSize; r++) {
        for (int c = 0; c < chessSize; c++) {
            chessboard[r][c] = ' ';
        }
    }

    for (int r = 0; r < chessSize; r++) {
        for (int c = 0; c < chessSize; c++) {
            if (r == 0 || r == 7) {
                if (c == 0 || c == 7) {
                    int knightPlacement,bishopPlacement;
                    int queenPlacement = 3;
                    int kingPlacement = queenPlacement + 1;

                    if (c == 0) {
                        knightPlacement = c+1;
                        bishopPlacement = knightPlacement+1;

                    }
                    else {
                        knightPlacement = c-1;
                        bishopPlacement = knightPlacement-1;
                    }

                    chessboard[r][c] = 'R';
                    chessboard[r][knightPlacement] = 'N';
                    chessboard[r][bishopPlacement] = 'B';
                    chessboard[r][queenPlacement] = 'Q';
                    chessboard[r][kingPlacement] = 'K';
```



```cpp

                }

                if (r==0){
                    chessboard[r+1][c] = 'P';
                }
                else {
                    chessboard[r-1][c] = 'P';
                }
            }
        }
    }

    for (int r = 0; r < chessSize; r++) {
        for (int c = 0; c < chessSize; c++) {
            if (c == 7) {
                std::cout << chessboard[r][c] << std::endl;
            } else {
                std::cout << chessboard[r][c];
            }
        }
    }

    return 0;
}
```

Output



```
RNBQKBNR
PPPPPPPP




PPPPPPPP
RNBQKBNR

--------------------------------
Process exited after 0.3326 seconds with return value 0
Press any key to continue . . .
```

7. Supplementary Activity

1.
First, the code initializes an int variable named int. Next, it declares a string array named days with an array size of seven. This array has values of a string of names of the days in a week. It prints “Enter an input” and waits for an input from the user that will be stored in the input variable. Then, an if statement checks if the input is greater than or equal to 0 but less than 7. If it is true then it prints the string value on the array days with the index inputted by the user. If it is not, then it prints “Error, no such day exists”. Lastly it returns 0.
2.
First, the code defines chessSize as 8. Then it creates a 2d char array named chessboard that has 8x8 size. The first for loop sets all of the values in the array with a blank space char. The next for loop has 2 parts. The outer part loops through the rows of the array. The inner part loops through the columns. The first if statement checks for the column number 0 and number 7, being the left and right. Then, it initializes the variables knightPlacement,bishopPlacement and declares queenPosition with the value of 3 and kingPosition with the value of queenPosition + 1 It then goes through another if statement that checks if it is at column 0, meaning it is on the left. If this statement is true then the knight will be placed on the right of the rook but if not, it will be placed on the left. The same logic applies to the bishopPlacement but instead of being relative to the rook it is relative to the knight. Line 31 - 35 places the pieces on the board. The next “if” statement checks if it is at row 1 (0). If it is, then it places it in front of them, at row 2(1). If not it places them at row 7(6). The last for loop also has an outer and inner for loop pretty similar to the last loop but this one just prints the content of the chessboard array.

8. Conclusion

I learned from the discussion part of the activity different sorting methods. First is the bubble sort, it is done by repeatedly swapping the adjacent values if they are in the wrong order. It is only suitable for sorting small arrays. Also, I learned about linear and binary search. On linear search, It checks the array from left to right using a loop to find the value inputted by the user. It is also only suitable for small arrays because of its slow nature if used on a big array. Binary search on the other hand is used for big arrays. It starts at the middle of the array. If the index of the given value is found then it stops. If it is not it checks if the given value is greater than or less than the middle value. If it is greater then it searches to the right, else it checks the left. It is suitable for larger arrays because it cuts the searching in half but it needs the array to be sorted. For the output, the first number of the activity is not that hard because it does not require any complex instruction. It just checks if the input number is on the range of the array size then prints accordingly. The second part on the other hand is pretty complex because it touches on the concept of 2 dimensional arrays. Imagining a chessboard helped in creating this code. The first iteration of the code has the problem of overwriting the parts of the other pieces except the rook leading to the rook only being printed.It took me time to realize that I needed to split it into different loops to properly print the desired output. The code analysis is not hard to do if you understand the code and how each part of it works. So, this part is not hard but only long to do because of the length of the codes.Overall, I learned and applied the different array searching method and the concept of 2d array but I think I still need to improve my understanding of it. This concept on 2d arrays is still new to me so there is still more to learn on this topic.

9. Assessment Rubric