EGRE 531 Multicore and Multithread Programming

Laboratory Number 2

Date: 02/16/18

Luis Barquero

PLEDGE: _____

"On my honor, I have neither given nor received unauthorized aid on this assignment"

VCU | Virginia Commonwealth University

Introduction:

The purpose of this lab was to create a method that would take in a string and a character, and the program will produce the formation of the letters with the character specified. That method will then be placed in two example codes provided that will simulate multiple threads and multiple processes with the method.

Lab Content:

First, all 26 letters and the numbers 0 to 9 were modeled in a Notepad++ file, as shown in Figure 1. Wherever a '*' occurs, it gets the value of 1, and wherever there is a space, it gets the value of 0. From there, five arrays were created for each line of all the characters to visually test each letter. An example of this is in Figure 2. Finally, all the 1s and 0s for each letter gets placed into one comprehensive array, as shown in Figure 3, and this is repeated for all five lines.

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

Figure 1 – Figure 1 is an example of how the program takes in a letter, in this case T, and models its shape with the corresponding character provided, which in this case is an asterisk.

```
int t1[10] = { 1,0,1,0,1,0,1,0,1,0 };
int t2[10] = { 0,0,0,0,1,0,0,0,0,0 };
int t3[10] = { 0,0,0,0,1,0,0,0,0,0 };
int t4[10] = { 0,0,0,0,1,0,0,0,0,0 };
int t5[10] = { 0,0,0,0,1,0,0,0,0,0 };
```

Figure 2 – Figure two shows the five arrays that represent the five lines to model the letter in the string. In this case, the letter T is diagrammed.

```
static int a1[317]
={0,0,1,0,1,0,0,0,1,0,1,0,1,0,1,0,0,0,1,0,1,0,1,0,1,0,1,0,1,0,0,0,0,1,0,1,0,1,0,1,0,1
,0,1,0,1,0,1,0,0,0,1,0,1,0,1,0,1,0,0,0,0,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,
0,0,0,1,0,1,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,1,0,1,0,0,0,0,0,0,0,1,0,0,0,1,0,1,0,1,0,0,0,1,0
,1,0,1,0,0,0,0,0,1,0,1,0,1,0,0,0,1,0,1,0,1,0,1,0,0,0,0,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,
1,0,0,0,0,0,1,0,1,0,0,0,0,0,1,0,1,0,0,0,0,0,0,0,1,0,1,0,0,0,0,0,0,0,1,0,1,0,0,0,0,0,0,0,1
,0,1,0,1,0,1,0,1,0,1,0,0,0,1,0,1,0,1,0,0,0,0,0,0,0,1,0,0,0,0,0,1,0,1,0,1,0,0,0,1,0,1,0,1,
0,0,0,1,0,0,0,0,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,0,0,1,0,1,0,0,0,1,0
,1,0,1,0,1,0 };
```

Figure 3 – Figure 3 shows an example of one of the final five arrays which mixes in all the letters and numbers.

Next, the width of each letter is calculated in order to make sure when a letter is accessed, only the correct amount is printed, and a pointer to all the five lines is created so that the for loop to print out the string access all five lines.

Finally, the loop to print out the string is in Figure 4. The loop goes through each of the five arrays and subtracts the character in the string with the ASCII value of ‘A’ to find begin modeling the correct letter. From there, if sums up all the arrays to determine the correct location of the letter, and once it finds the starting point, the next line sets the condition to place the character provided where there is a ‘1’ from the starting point to its ending location. If there is a ‘0’ from the starting point to the ending point, then the program will produce a space.

```
for (int l = 0; l < 5; l++) {
      cout << "\n";
          for (int i = 0; s[i] != '\0'; i++)
          {
             cout << " ";
             int ch = s[i] - 64;
             int si = 0;
                    for (int j = 0; j < ch - 1; j++) si += size[j];
                    for (int i = 0; i < size[i]; i = i + 1)
                    {
                       if (lines[l][si + i] == 1)
                      {
                         cout << c;
                       }
                       else
                      {
                          cout << " ";
                       }
                    }
          }
}
```

Figure 4 – Figure 4 shows the loop in the function that will go through every letter in the string and diagram each letter with the corresponding character.

This program was then placed in the two example programs provided to simulate how multithreading and multiprocessing works.

For the multithreaded case, the first string to output is “ECE” with the “*” symbol in the first process.
Next, the thread waits 10 seconds and the second output is “LRB” with the “*” symbol, still in the same thread. Finally, the second process is modified to print out “531” with the “#” character.

For the multi-process case, the first string is the same, with “ECE” and the “*” character, except this is placed in the character process. Next, the process waits 10 seconds and in the same process, the output “LRB” with the “*” is printed to the terminal. Finally, in the parent process, the string “531” with the character is “#” is outputted.

Test Results:

Multithreaded:

For the multithreaded program, Figure 5 shows the terminal output of the program, in which three strings and two characters are passed to two different threads.
The first output should be “531” with the character “*,” but because of the spacing in the layout of the letters and the loop condition that subtracts ‘A’ from every letter, it is outputted as “AAA.”
The next output is “ECE” with the “*”, but once again, because of the spacing in the layout of each letter, it adds the beginning of the first letter in the mix.
Finally, the last output is “LRB” with the character “*,” with the added beginning of the next letter.

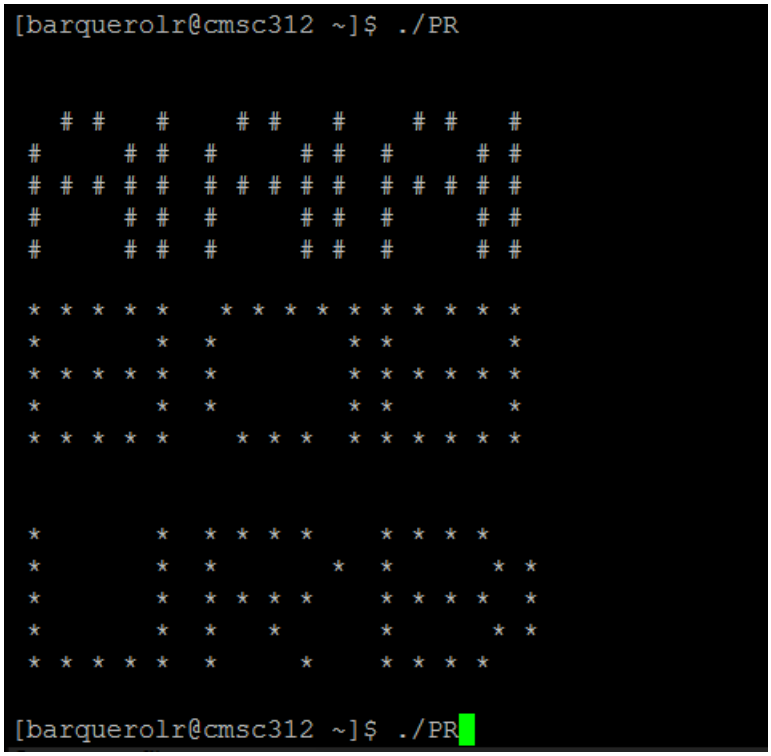

Figure 5 – Figure 5 shows the output of the multithreaded program, with “AAA (instead of 531) displayed with the “#” character, “ECE” with the “*” character, and finally “LRB” with the “*” character.

Multi-process:

Figure 6 shows the output of the multi-process example. Similar to the to multithreaded example, there are three strings outputted, each with their own specified character. The first string is “AAA”(should be “531”) with the character “#” the second string is “ECE” with the “*” character and the final string of “LRB” with the “*” character.

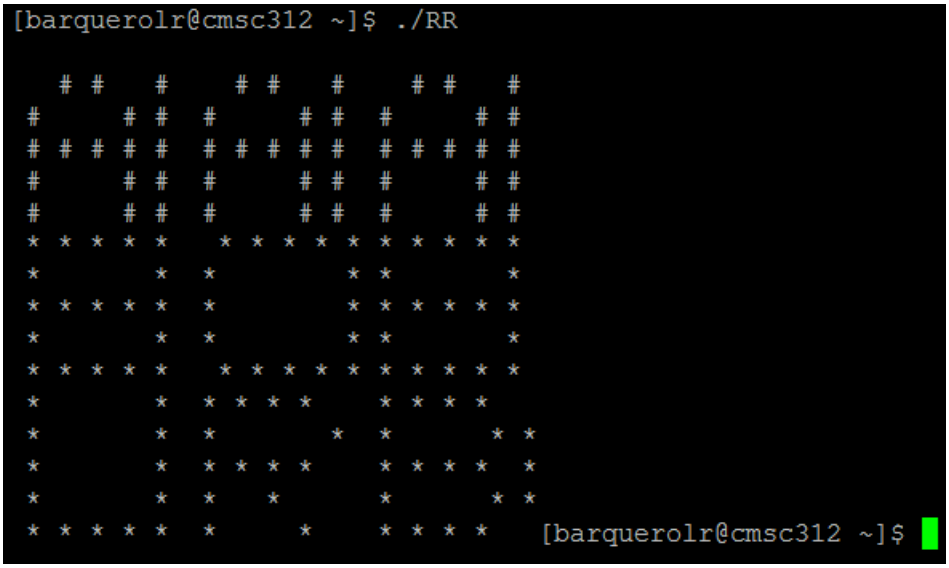

Figure 6 – Figure 6 shows terminal output for the multi-process program. The first string is "AAA (instead of 531) displayed with the "#" character, then "ECE" with the "*" character, and finally "LRB" with the "*" character.

Analysis:

From both programs, the output was the same, with the "531" string displayed first, then the "ECE" string and finally the "LRB" string. The only difference is that the multi-threaded program added an extra line between each new string.

After watching other classmates' results, this output is wrong. It should output the "ECE" string first, then the "531" string, and it should finish with the "LRB" string. This is because the "ECE" is in thread one, and since "531" only has to wait for 3 seconds, it's outputted second, while "LRB" printed last since it waits the most time, 10 seconds.

In the multi-process method, it's the same idea, since the condition says "if pid == 0," meaning there has not been a parent process created yet, so it should print out "ECE" first. Then, "531" should be printed next, since now there is a parent function and "531" only waits 3 seconds. Finally, "LRB" comes last since the child process has to wait 10 seconds to print out.

The letter formation also change after every execution of the program.

Problems Encountered:

The two main problems encountered were the strings not outputted properly because of the spacing in the letters and the strings outputted in the correct order. As a result, the program outputs the correct letter but adds the beginning of the next letter, and the strings are not in order.

Appendix A

Source Code: Multi-Thread

```
/*******************************************************************************
EGRE 531 Lab 2
Programmed by: Luis Barquero
Purpose: Passes in a string and character, and models every letter in the string
         with the corresponding character. This is for the multithreaded case.
*******************************************************************************/
#include <iostream>
#include <pthread.h>
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>

using namespace std;
void *thread1(void *);
void *thread2(void *);
void print_string(char*, char);

char *specified_character;

int main(void)
{
    char string_input[32];
    char *buffer = string_input;
    pthread_t t_a;
    pthread_t t_b;

    //Add your codes here to allocate memory space from the heap here
    buffer = (char*)malloc(sizeof(string_input));

    pthread_create(&t_a, NULL, thread1, (void *)NULL);
    pthread_create(&t_b, NULL, thread2, (void *)NULL);
    pthread_join(t_a, NULL);
    pthread_join(t_b, NULL);

    free(buffer);

    exit(0);

}

void *thread1(void *junk)
{
    //Add your codes here to display different strings according to the description
    print_string("ECE", '*');
    unsigned int seconds = 10;
    unsigned int sleep(seconds);
    print_string("LRB", '*');
}

void *thread2(void *junk)
{
    unsigned int seconds = 3;
    unsigned int sleep(seconds);
    //Add your codes here to display different strings according to the descriptions
    print_string("531", '#');
}

void print_string(char* s, char c)
{
    static int a1[317] = { 0,0,1,0,1,0,0,0,1,0,1,0,1,0,1,0,0,0,1,0,1,0,1,0,1,0,1,0,1,0,1
    ,0,0,0,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,0,0,1,                          0,1,0,1,0,
                            1,0,0,0,0,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,0
                            ,0,1,0,1,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,1,0,
                            1,0,0,0,0,0,0,0,1,0,0,0,1,0,1,0,1,0,0,0,1,0,1,0,1,0,0,0,0,1
                            ,0,1,0,1,0,0,0, 1,0,1,0,1,0,1,0,0,0, 0,0,1,0,1,0,1,0,
                            1,0,1,0,1,0,1,0,1,0,1,0,0,0,0,0,1,0,1,0,0,0,0,1,0,1,0,0,0,0
                            ,0,0,0,1,0,1,0,0,0,0,0,0,0,1,0,1,0,0,0,0,0,0,0,1,0,
                            1,0,1,0,1,0,1,0,1,0,1,0,0,0,1,0,1,0,1,0,0,0,0,0,0,0,0,1,0,0,0,0,0,1
                            ,0,1,0,1,0,1,0,0,0,1,0,1,0,1,0,1,0,0,0,1,0,0,0,0,0,1,0,
```

```
                        1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,0,0,1,0,1,0,0
                        ,0,1,0,1,0,1,0,1,0};

    static int a2[317] = { 1,0,0,0,0,0,1,0,1,0,0,0,0,0,0,1,0,1,0,0,0,0,0,0,0,1,0,0,0,0
    ,0,0,1,0,1,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,1,0,0,                    0,0,0,0,0,
                        1,0,0,0,0,0,0,1,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,1,0,1
                        ,0,0,0,1,0,0,0,0,0,0,0,1,0,1,0,0,0,1,0,1,0,
                        1,0,1,0,0,0,0,0,1,0,1,0,0,0,0,0,0,0,1,0,1,0,0,0,0,0,1,0,1,0,0
                        ,0,0,0,0,0,0,1,0,1,0,0,0,0,0,0,0,1,0,1,0,0,0,0,0,0,0,0,0,0,
                        0,0,0,0,1,0,0,0,0,0,1,0,0,0,0,0,0,1,0,1,0,0,0,0,0,0,1,0,1,0,0,0
                        ,0,0,0,1,0,0,0,1,0,0,0,1,0,0,0,0,0,1,0,0,0,1,0,0,0,
                        0,0,0,0,0,0,0,1,0,0,0,1,0,0,0,0,0,0,0,1,0,0,0,1,0,1,0,0,0,0,0,0
                        ,0,0,0,0,0,0,1,0,0,0,0,0,0,0,1,0,1,0,0,0,0,0,1,0,
                        1,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,1,0,0,0,0,0,1
                        ,0,1,0,0,0,0,0,1,0,1,0,

    static int a3[317] = { 1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,0,1,0,0,0,0,0,0,0,1,0,0,0,0
    ,0,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,                    0,1,0,1,0,
                        1,0,1,0,1,0,1,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,1,0,0
                        ,0,0,0,1,0,0,0,0,0,0,0,1,0,0,0,1,0,0,0,1,0,
                        1,0,0,0,1,0,0,0,1,0,1,0,0,0,0,0,0,0,1,0,1,0,1,0,1,0,0,0,1,0,0
                        ,0,0,0,0,0,0,1,0,1,0,1,0,1,0,1,0,0,0,1,0,1,0,1,0,0,0,0,
                        0,0,0,0,1,0,0,0,0,0,1,0,0,0,0,0,1,0,1,0,0,0,0,0,1,0,1,0,0,0,1
                        ,0,0,0,1,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,1,0,0,0,0,
                        0,0,0,0,0,1,0,0,0,0,0,1,0,1,0,1,0,1,0,1,0,1,0,0,0,1,0,0,0,0,0,1
                        ,0,1,0,1,0,1,0,1,0,1,0,1,0,0,0,1,0,1,0,1,0,1,0,
                        1,0,1,0,1,0,0,0,1,0,1,0,1,0,1,0,0,0,0,0,0,0,1,0,1,0,0,0,0,0,1,0
                        ,0,0,0,0,0,0,0,1,0,

    static int a4[317] = { 1,0,0,0,0,0,1,0,1,0,0,0,0,0,1,0,1,0,0,0,0,0,0,0,1,0,0,0,0
    ,0,0,1,0,1,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,1,0,0,                    0,0,0,1,0,
                        1,0,0,0,0,0,0,1,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,1,0,1
                        ,0,0,0,1,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,1,0,
                        1,0,0,0,0,0,0,1,0,1,0,1,0,0,0,0,0,0,0,1,0,1,0,0,0,0,0,1,0,1,0,0
                        ,0,0,0,1,0,1,0,1,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,1,0,
                        0,0,0,0,1,0,0,0,0,0,1,0,0,0,0,0,1,0,1,0,0,0,0,0,1,0,1,0,1,0,0
                        ,0,1,0,1,0,0,0,1,0,0,0,1,0,0,0,0,0,1,0,0,0,0,0,
                        0,0,1,0,0,0,0,0,0,0,1,0,0,0,0,0,0,1,0,0,0,0,0,0,1,0,0,0,0,0,1
                        ,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,1,0,
                        0,0,0,0,0,0,1,0,1,0,0,0,0,0,1,0,0,0,0,0,0,0,1,0,1,0,0,0,0,0,1
                        ,0,0,0,0,0,0,0,1,0};

    static int a5[317] = { 1,0,0,0,0,0,1,0,1,0,1,0,1,0,1,0,0,0,0,1,0,1,0,1,0,0,1,0,1,0,1
    ,0,0,0,0,1,0,1,0,1,0,1,0,1,0,0,0,0,0,0,0,0,0,1,                    0,1,0,1,0,
                        1,0,0,0,0,0,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,0,0,0,0,1,0,0
                        ,0,1,0,1,0,1,0,1,0,1,0,1,0,0,0,0,0,0,0,1,0,
                        1,0,0,0,0,0,0,0,0,1,0,0,0,1,0,1,0,1,0,0,0,1,0,0,0,0,0,0,0,0,0,1
                        ,0,1,0,1,0,1,0,1,0,0,0,0,0,1,0,0,0,1,0,1,0,1,0,0,0,
                        0,0,0,0,1,0,0,0,0,0,1,0,1,0,1,0,1,0,0,0,1,0,1,0,0,0,1,0,0,0,0
                        ,0,0,0,1,0,1,0,0,0,0,0,0,0,1,0,0,0,0,0,1,0,0,0,0,0,
                        1,0,1,0,1,0,1,0,0,0,0,0,1,0,1,0,1,0,0,0,1,0,1,0,1,0,1,0,1,0,0
                        ,0,1,0,1,0,1,0,1,0,1,0,1,0,0,0,0,0,0,0,0,0,1,0,
                        1,0,1,0,1,0,0,0,1,0,1,0,1,0,1,0,0,0,0,0,0,0,1,0,0,0,1,0,1,0,0
                        ,0,0,0,0,0,0,0,1,0};

    static int *lines[] = { a1, a2, a3, a4, a5 };
    static int size[36] = { 8,9,9,9,8,8,8,8,10,10,6,8,10,10,10,8,10,10,8,10,8,8,10,10,10
    ,10,10,10,8,8,8,8,8,8,8,8};
    int i;

    for (int l = 0; l < 5; l++)
    {
        cout << "\n";
        for (int i = 0; s[i] != '\0'; i++)
        {
            cout << " ";
            int ch = s[i] - 64;
            int si = 0;
            for (int j = 0; j < ch - 1; j++) si += size[j];
```

```
            for (int i = 0; i < size[i]; i = i + 1)
            {
                if (lines[l][si + i] == 1)
                {
                    cout << c;
                }
                else
                {
                    cout << " ";
                }
            }
        }
        cout << "\n\n";
    };
```

Appendix B

Source Code: Multi-Process

```
/*****************************************************************************
EGRE 531 Lab 2
Programmed by: Luis Barquero
Purpose: Passes in a string and character, and models every letter in the string
         with the corresponding character. This is for the multiprocess case.
*****************************************************************************/
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <sys/types.h>
#include <iostream>
#include <unistd.h>
using namespace std;
void print_string(char*, char);

int main(void)
{
    char string_input[32];
    char *buffer = string_input;
    //Add your codes here to allocate memory space from the heap here
    buffer = (char*)malloc(sizeof(string_input));

    pid_t pid;
    pid = fork();
    if (pid == 0) //child process
    {
        //Add your codes here to display different strings according to the descriptions
        print_string("ECE", '*');
        unsigned int seconds = 10;
        unsigned int sleep(seconds);
        print_string("LRB", '*');
    }
    else //parent process
    {
        unsigned int seconds = 3;
        unsigned int sleep(seconds);
        print_string("531", '#');
        ////Add your codes here to display different strings according to the
        descriptions
    }
    free(buffer);

    return 0;
}

void print_string(char* s, char c)
{
    static int a1[317] = { 0,0,1,0,1,0,0,0,1,0,1,0,1,0,1,0,0,0,1,0,1,0,1,0,1,0,1,0,1,0,1
    ,0,0,0,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,0,0,1,0,1,0,1,0,
        1,0,0,0,0,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,0,0,1,0,1,0,0,0,0,0,
        0,0,1,0,0,0,0,0,0,0,1,0,
        1,0,0,0,0,0,0,0,1,0,0,0,1,0,1,0,1,0,0,0,1,0,1,0,1,0,0,0,0,0,1,0,1,0,1,0,0,0,1,0,
        1,0,1,0,1,0,1,0,0,0,0,0,1,0,1,0,1,0,
        1,0,1,0,1,0,1,0,1,0,0,0,0,0,0,1,0,1,0,0,0,0,0,1,0,1,0,0,0,0,0,0,0,1,0,1,0,0,0,
        0,0,0,0,1,0,1,0,0,0,0,0,0,0,0,1,0,
        1,0,1,0,1,0,1,0,1,0,0,0,1,0,1,0,1,0,0,0,0,0,0,0,1,0,0,0,0,0,1,0,1,0,1,0,0,0,1,0,
        1,0,1,0,0,0,1,0,0,0,0,0,1,0,
        1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,0,0,1,0,1,0,0,0,1,0,1,0,1,0,1,0
        };

    static int a2[317] = { 1,0,0,0,0,0,1,0,1,0,0,0,0,0,0,1,0,1,0,0,0,0,0,0,0,0,1,0,0,0,0
    ,0,0,1,0,1,0,0,0,0,0,0,0,1,0,0,0,0,0,0,1,0,0,0,0,0,0,0,
        1,0,0,0,0,0,1,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,1,0,1,0,0,0,1,0,0,0,0,0,
        0,0,1,0,1,0,0,0,1,0,1,0,
        1,0,1,0,0,0,0,0,1,0,1,0,0,0,0,0,0,0,1,0,1,0,0,0,0,0,1,0,1,0,0,0,0,0,0,0,1,0,1,0,
        0,0,0,0,0,0,1,0,1,0,0,0,0,0,0,0,
        0,0,0,0,1,0,0,0,0,0,1,0,0,0,0,0,1,0,1,0,0,0,0,0,1,0,1,0,0,0,0,0,0,0,1,0,0,0,1,0,
        0,0,1,0,0,0,0,0,1,0,0,0,1,0,0,0,
        0,0,0,0,0,0,1,0,0,0,1,0,0,0,0,0,0,0,1,0,0,0,1,0,1,0,0,0,0,0,0,0,0,0,0,1,0,0,0,
```

```
        0,0,0,0,1,0,1,0,0,0,0,0,1,0,
59      1,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,1,0,0,0,0,0,1,0,1,0,0,0,0,0,1,0
        };
60
61      static int a3[317] = { 1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,0,1,0,0,0,0,0,0,0,1,0,0,0,0
        ,0,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,
62      1,0,1,0,1,0,1,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,1,0,0,0,0,0,1,0,0,0,0,0,
        0,0,1,0,0,0,1,0,0,0,1,0,
63      1,0,0,0,1,0,0,0,1,0,1,0,0,0,0,0,0,0,1,0,1,0,1,0,1,0,0,0,1,0,0,0,0,0,0,0,1,0,1,0,
        1,0,1,0,1,0,0,0,1,0,1,0,1,0,0,0,
64      0,0,0,0,1,0,0,0,0,0,1,0,0,0,0,0,1,0,1,0,0,0,0,0,1,0,1,0,0,0,1,0,0,0,1,0,0,0,0,0,
        1,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,
65      0,0,0,0,1,0,0,0,0,0,1,0,1,0,1,0,1,0,1,0,1,0,0,0,1,0,0,0,0,0,1,0,1,0,1,0,1,0,1,0,
        1,0,1,0,0,0,1,0,1,0,1,0,1,0,
66      1,0,1,0,1,0,0,0,1,0,1,0,1,0,1,0,0,0,0,0,0,0,1,0,0,0,1,0,1,0,0,0,1,0,1,0,1,0,1,0
        };
67
68      static int a4[317] = { 1,0,0,0,0,0,1,0,1,0,0,0,0,0,1,0,1,0,0,0,0,0,0,0,1,0,0,0,0
        ,0,0,1,0,1,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,1,0,0,0,0,0,1,0,
69      1,0,0,0,0,0,1,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,1,0,0,0,0,0,1,0,1,0,0,0,1,0,0,0,0,0,
        0,0,1,0,0,0,0,0,0,0,1,0,
70      1,0,0,0,0,0,1,0,1,0,1,0,0,0,0,0,0,0,1,0,1,0,0,0,0,0,0,0,1,0,0,0,0,0,1,0,1,0,1,0,
        0,0,1,0,0,0,0,0,0,0,0,0,0,1,0,
71      0,0,0,0,1,0,0,0,0,0,1,0,0,0,0,0,1,0,1,0,0,0,0,0,1,0,1,0,1,0,0,0,1,0,1,0,0,0,1,0,
        0,0,1,0,0,0,0,0,0,0,1,0,0,0,0,
72      0,0,1,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,1,0,0,0,0,0,1,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,
        0,0,0,0,1,0,0,0,0,0,0,0,1,0,
73      0,0,0,0,0,0,1,0,1,0,0,0,0,0,1,0,0,0,0,0,0,0,1,0,1,0,0,0,0,0,1,0,0,0,0,0,0,0,1,0
        };
74
75      static int a5[317] = { 1,0,0,0,0,0,1,0,1,0,1,0,1,0,1,0,0,0,1,0,1,0,1,0,1,0,1,0,1,0,1
        ,0,0,0,0,1,0,1,0,1,0,1,0,1,0,0,0,0,0,0,0,0,0,0,1,0,1,0,1,0,
76      1,0,0,0,0,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,0,0,0,0,1,0,0,0,1,0,1,0,1,0,1,0,
        1,0,1,0,0,0,0,0,0,0,1,0,
77      1,0,0,0,0,0,0,0,1,0,0,0,1,0,1,0,1,0,0,0,1,0,0,0,0,0,0,0,0,0,1,0,1,0,1,0,1,0,1,0,
        0,0,0,0,1,0,0,0,1,0,1,0,1,0,0,0,
78      0,0,0,0,1,0,0,0,0,0,1,0,1,0,1,0,1,0,0,0,1,0,1,0,0,0,1,0,0,0,0,0,0,0,1,0,1,0,0,0,
        0,0,0,0,1,0,0,0,0,0,1,0,0,0,0,0,
79      1,0,1,0,1,0,1,0,0,0,0,0,1,0,1,0,1,0,0,0,1,0,1,0,1,0,1,0,1,0,0,0,1,0,1,0,1,0,1,0,
        1,0,1,0,0,0,0,0,0,0,0,0,1,0,
80      1,0,1,0,1,0,0,0, 1,0,1,0,1,0,1,0,0,0,0,0,0,0,1,0,0,0,1,0,1,0,0,0,0,0,0,0,0,0,1,0
         };
81
82      static int *lines[] = { a1, a2, a3, a4, a5 };
83      static int size[36] = { 8,9,9,9,8,8,8,8,10,10,6,8,10,10,10,8,10,10,8,10,8,8,10,10,10
        ,10,10,10,8,8,8,8,8,8,8,8 };
84      int i;
85
86      for (int l = 0; l < 5; l++) {
87          cout << "\n";
88          for (int i = 0; s[i] != '\0'; i++)
89          {
90              cout << " ";
91              int ch = s[i] - 64;
92              int si = 0;
93              for (int j = 0; j < ch - 1; j++) si += size[j];
94              for (int i = 0; i < size[i]; i = i + 1)
95              {
96                  if (lines[l][si + i] == 1)
97                  {
98                      cout << c;
99                  }
100                 else
101                 {
102                     cout << " ";
103                 }
104             }
105         }
106     }
```

```
107   };
```