Advanced Operating Systems

Assignment #1

Seyed Javad Khataei Pour

201793290

October2018

# Defining the problem.

This assignment has three subproblems.

## Subproblem one-Random generator problem.

subproblem one is designing a routine that can produce random integer numbers within the range of 1 to a limit. The user should be able to give the function the range an also the seed. Moreover, there should be an option which the function uses system clock as the seed. In this manner the randomness of the system will increase. At the end this function will produce a sequence of random integer numbers and displays it on the console.

Second subproblem is designing three similar functions which print letters A, B ,and C. This function will be used in the next subproblem.

The third subproblem combines previous subproblems and each time a random number is generated from subproblem one, one of the functions of subproblem 2 will invoke and print either letter A, B or C.

# solution

## Top-to-bottom design

For the first step, the problem will be broken down to smaller pieces up to the point that each subproblem is easy to understand and convenient to solve.

### Subproblem one-Random generator problem.

for this section the user has three choices.

First option is giving only the range. In this case infinite numbers of random integers will be produced within the range. Here system clock is used as the seed.

Second option: user gives the seed and also the range so infinite numbers of random integers will be produced within the range and based on the provided seed by the user.

Third option: The user will provide the seed, range and also the quantity of random numbers that need to be produced. The output in this case will be a finite sequence of random integers within the given range based on the provided seed.

Main software structure: in this section an interface is designed to provide the user an user-friendly, easy to work with environment. user can easily navigate through different part of the software and execute different sub-functions regarding three aforementioned subproblems. Moreover, user can use the option menu to manipulate each function. More details about the entire structure is provided in the next chapter.

## Top-to-bottom

In order to make the problem easier to solve top-to-bottom method is used and the main problem in divided to several subproblems.

### Subproblem #1:

First subproblem is a random generator. Although it might seem to be easy to implement, this subproblem also needs to be divided into smaller parts. This subproblem itself has four subproblems. This subproblems can be called subproblem1.1 , subproblem1,2 , etc.

subproblem 1.1:

The first one is a problem that uses the system clock as the seed for the rand() function. It gets the range and provide infinite numbers of random integers between 1 and the limit that user provides here.

Subproblem 1.2:

Second subproblem of subproblem one gets the range and a seed from the user and use them to generate infinite number of integers within the given range similar to subproblem1.1.

subproblem 1.3:

This routine produce a certain number of random integers. The user decides how many random number should be produced. Also user provide a range. For this function the seed is coming from system clock.

Subproblem 1.4:

this function is similar to subproblem 1.3. The only difference is that the user provide a seed and that seed will be used for random generation.

### Subproblem #2:

This subproblem has three similar routines. The first one simply prints out the letter “A”. The second and the third one print out letter”B” and “C” respectively.

Subproblem#3:

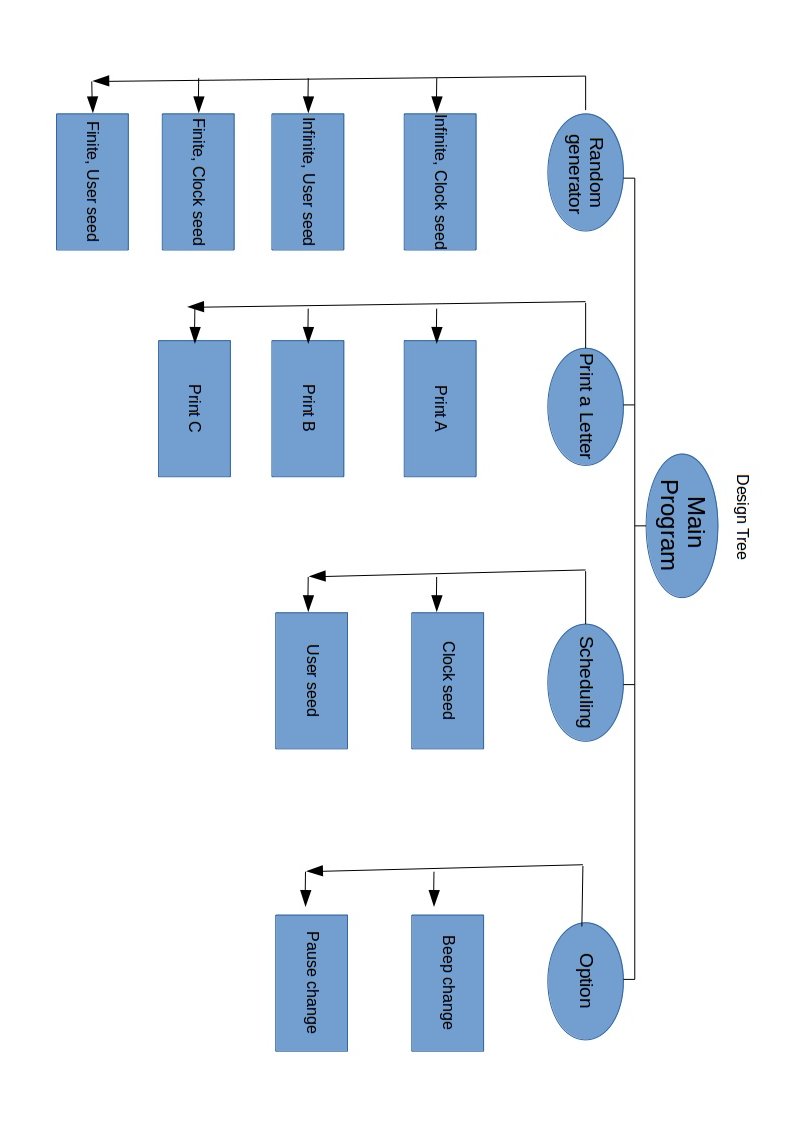
In this section subproblem 1 and 2 are combined and based on the value generate by the subproblem one, one of the functions of the subproblem 2 will invoke and prints out a sequence of random letters. This subproblem is devided into two part.

Subproblem 3,1:

This function produces infinite number of integer based on a seed from the system clock. Moreover it uses the random number to print out “A”, “B”, or “C”.

Subproblem 3,2:

This function get a seed from the user and then according to that seed produces infinite number of integer. Also, it uses the random number to print out “A”, “B”, or “C”.



### Programing language:

For this assignment C language is used. Eclipse idea were used to write and compile the code on a windows 10 system. The reason that C is used is that this language is close to machine language so the speed is very satisfying. The second reason is that XINU is written in C, therefore by learning C XINU will be more understandable. And finally writing the program in C was more challenging and demanding comparing to other languages thus the learning outcome would be greater.

# Code structure:

In order to keep the code readable and tidy, all the functions are in another file called functions( except the main function). Moreover for each task a function is written to keep the code easy to expand and well structured. Most of the function can be divided into to main group. First group is responsible for displaying the menu and helping the user to navigate and choose the desirable function. The second type is the functions that calculate or produce a set of results. In each category there are several kinds of functions each doing an specific job. However they can fit to certain categories. The following diagram shows function types hierarchy. The function naming has dome carefully to increase the code readability.

## Structural functions:

There are three different types:

### Navigation handler:

This function allows the user to navigate through different option of a menu and exit the menu by pressing ESC key. It also keep track of the active option of the menu and sends it to other functions(Display and luncher) to invoke appropriate computational functions.

### Display:

This type of function provide an user-friendly and pleasant environment for the user. Setting the color of the text and viewing the menu is done by this type of functions

### Luncher:

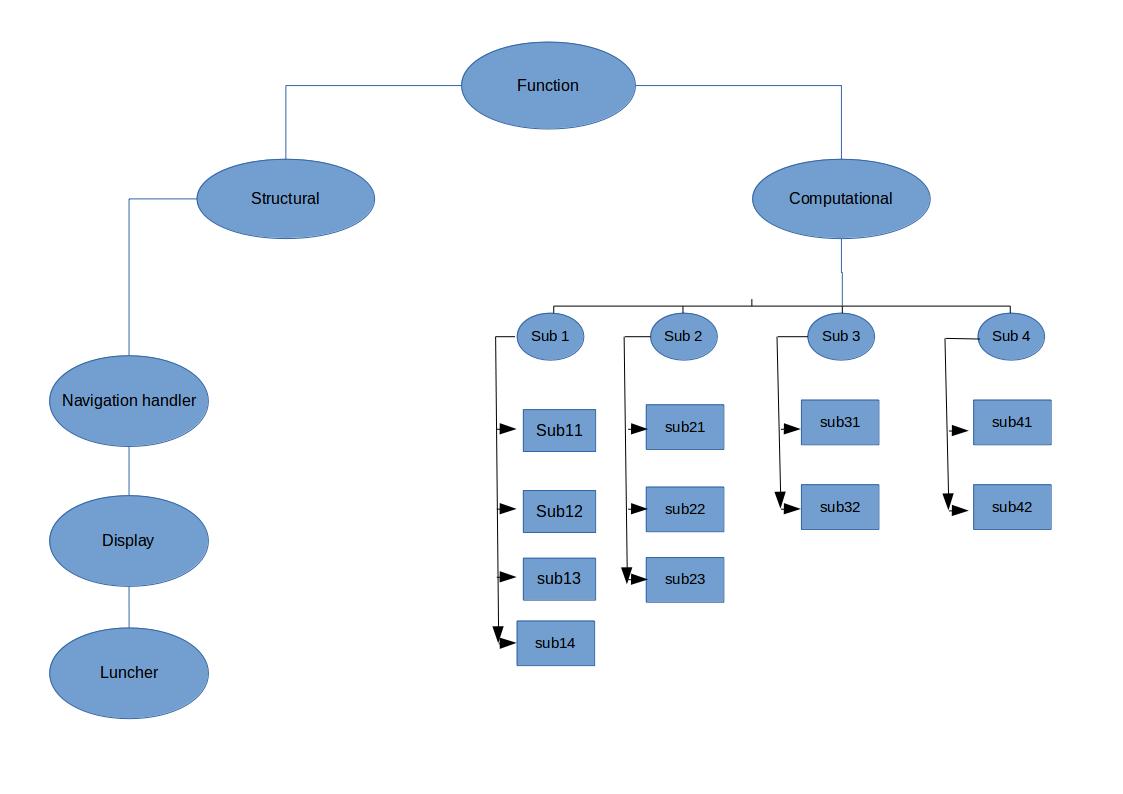
When the user finalize his or her chose by pressing the enter button the navigation handler invoke this type( luncher) and the luncher then invoke one of the computational functions based on the user choice. As mentioned earlier user choice is captured by navigation handler and passed to the luncher.

Since there are several menus the are a number of each type. The naming is meant to be straight forward and self explanatory. For example, the function lunchsub2 is a luncher for the second subproblem. Or sub1Menu() is the navigation handler of subproblem one’s menu. The function displaysub3Menu is responsible for viewing the menu of subproblem #3;

### Computational functions

here computational function are the ones which calculate or print out a value. For example routines of subproblem two which prints A, B, or C are computational functions.

The naming is based on the subproblem number. For example sub13 points to the function that solve subproblem 1 part three. In general the naming of this function follows the logic behind the presented design tree.



# Code explanation:

the code is complemented with fair amount of comments to make it easy to understand. Since there are a few type of function, understanding each from one type will be helpful in analyzing the other function from the same type. In this section as an example one the functions will be explained throughly.

Name:sub1Menu

Task: handles sub menu #1.

Description: Watches for the keys user presses and control navigation

void sub1Menu(){

int key1,sub1ChoiceCounter;

//sub1ChoiceCounter is the user choice for the subproblem 1 menu. this is is set to 101 as a base so when user presses the Up button and mainChoiceCounter decrease it will be remain positive number which is easier to handle here. Note that % operator is used here so for negative numbers extra correction is required. Setting the base to 100 is an effort to avoid extra work

//key1 is the variable that contains the last pressed key. Then it can be send to sub-function. However in the program this has been done only once in the main menu.

Using this key to going back to the upper menu is possible however this feature is not used except for the main menu because it is easier for user to press ESC key and navigate to upper menu rather than selecting an Exit option. For the main menu this key variable was used to close the program in order to prevent accidental closing by pressing ESC while browsing the main menu

sub1ChoiceCounter=101;

displaysub1Menu(sub1ChoiceCounter);

// This function display the main menu based on the user's choice which is stored in sub1ChoiceCounter variable as an integer

this while loop keep the menu active up to the point that the user presses ESC and sends 27 to key1 variable. In that case while loop ends

while ( key1 != 27)

{

key1= getch();// reading the user's key pressed

if (key1==80)// The Down arrow key is pressed

{

sub1ChoiceCounter++;

sub1ChoiceCounter=sub1ChoiceCounter%4+100;

//This keep the sub1ChoiceCounter in a certain range so we can assign different function for each choice easily by using if in the displaymainmenu() function

displaysub1Menu(sub1ChoiceCounter);

}

else if(key1==72)// The Up arrow key is pressed

{

sub1ChoiceCounter--;

sub1ChoiceCounter=(sub1ChoiceCounter%4+100) ;

displaysub1Menu(sub1ChoiceCounter);}

else if(key1==13)// The Enter arrow key is pressed.{

// Invoke different function based on the menu that is selected by the user lunchsub1(sub1ChoiceCounter,&key1);

// Sending by reference so we can edit the key pressed in case of need( eg. exit)

displaysub1Menu(sub1ChoiceCounter); }

}// End of the while loop

}//End of the function

Appendix A contains all the code which is slightly more than 1000 lines.

# Software

## Required system to use software:

This software is written in c language and the compiler GCC is used to compiled it. This program is developed and tested on window10. With slight justification the code can be compiled on Linux as well, however the executable file is for windows. It should run smoothly on any computers running a version of windows.

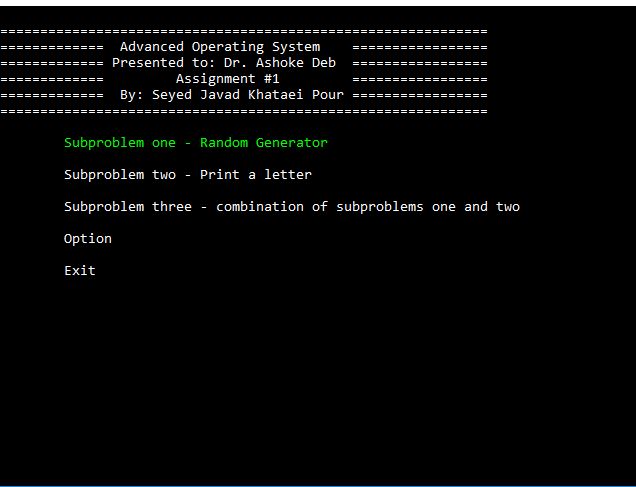
## How to work with the software:

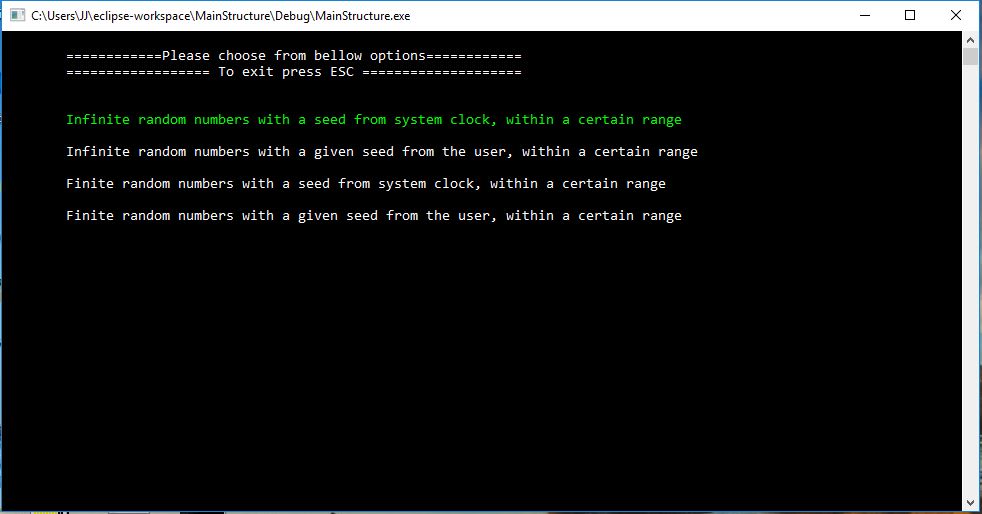
Although designing user-friendly user interference is not easy in C, a very straightforward and clear approach was used to develop this software. The user will be able to navigate through the menus by Up and Down keys. In case of selecting an option the Users presses the Enter key while the desirable option is active. The green line of the menu is the active one and the white ones are not selected. ESC key can be used to exit most menus except the main menu. In order to close the program Simply navigate to the Exit option in the main menu and press the Enter key. In the option menu, the user can modify the beep and the pauses time between each out put. Setting a beep and a short pause make the program elegant and professional.

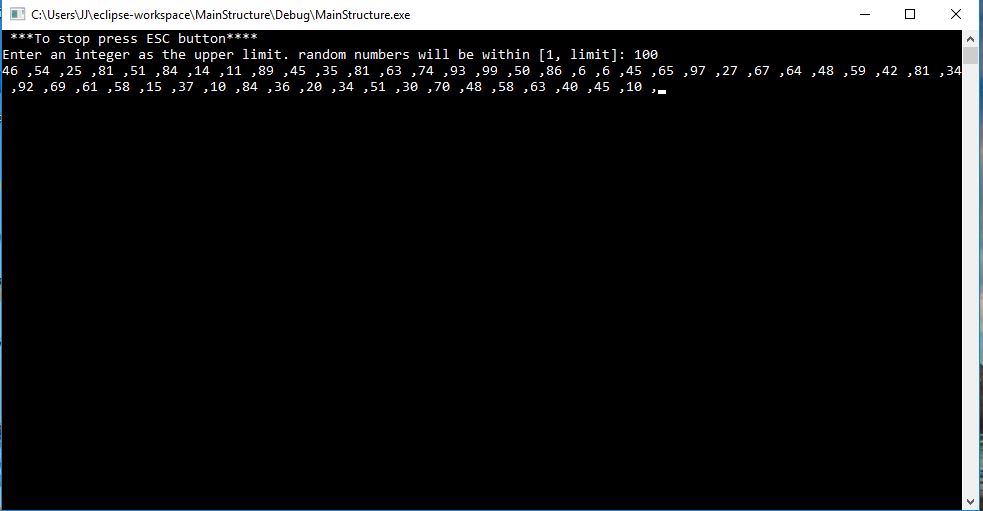
## Input and Output:

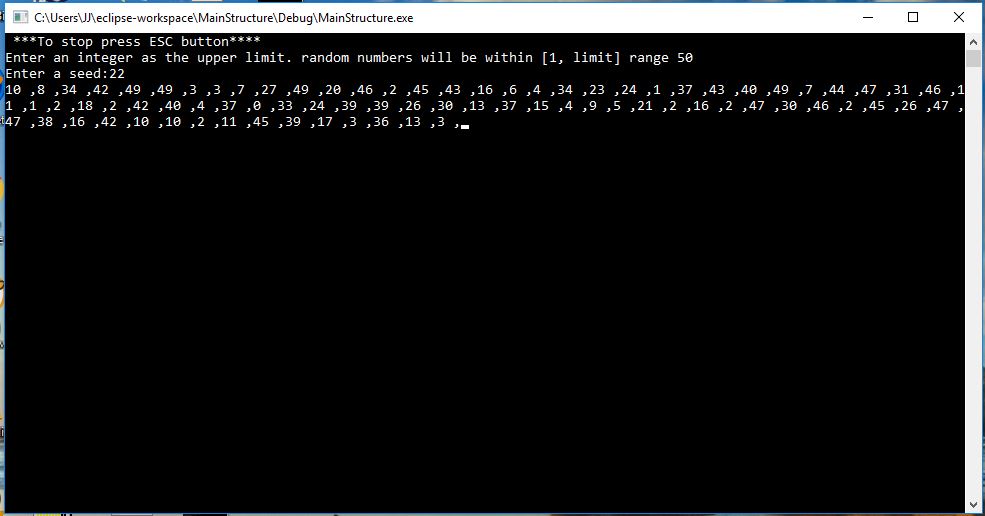
Inputs are all integers and mostly should be positive. Therefore if an negative value is entered by the user, the program will continue asking the user for appropriate values and does not proceed since it can cause a fatal error. In the following pictures for each subproblem sample inputs are provided.

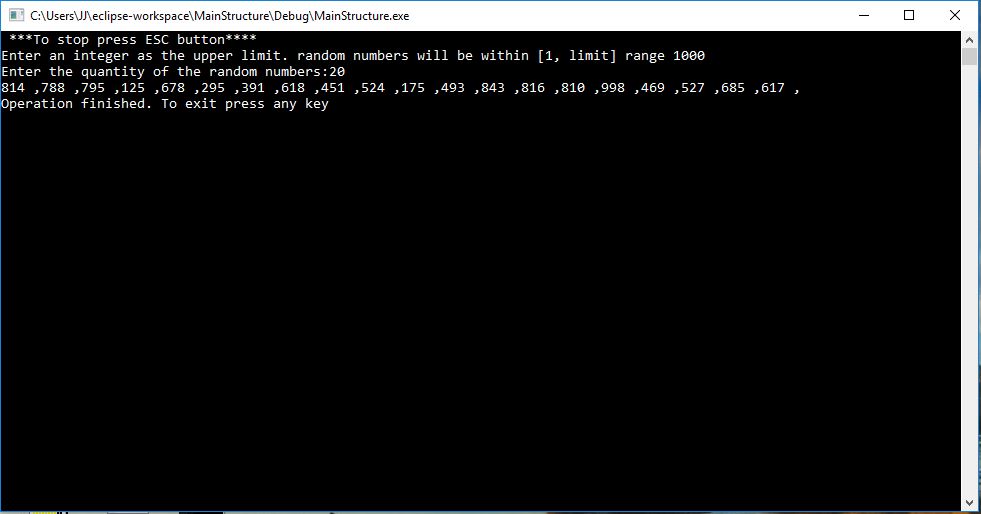
The output for each function is also presented in the snapshots.

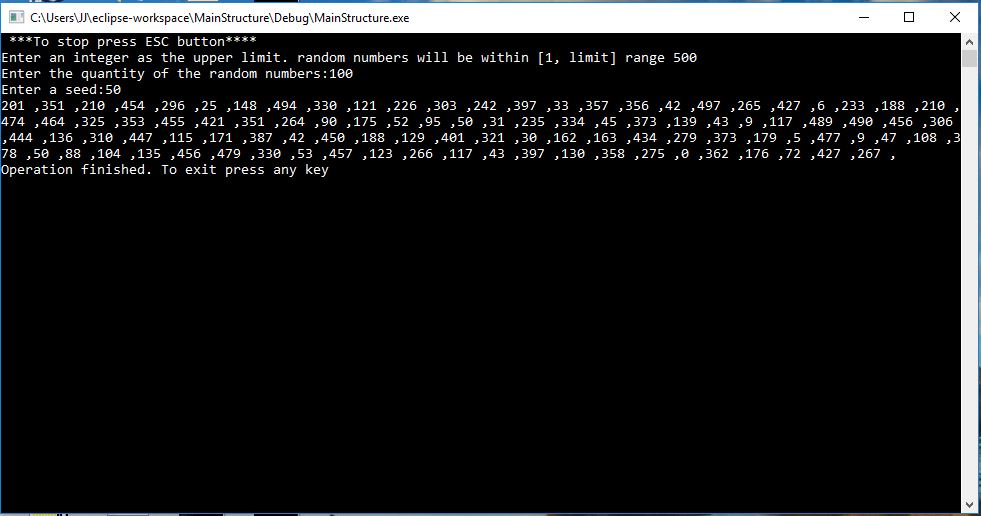
  
Illustration 1: The Main menu

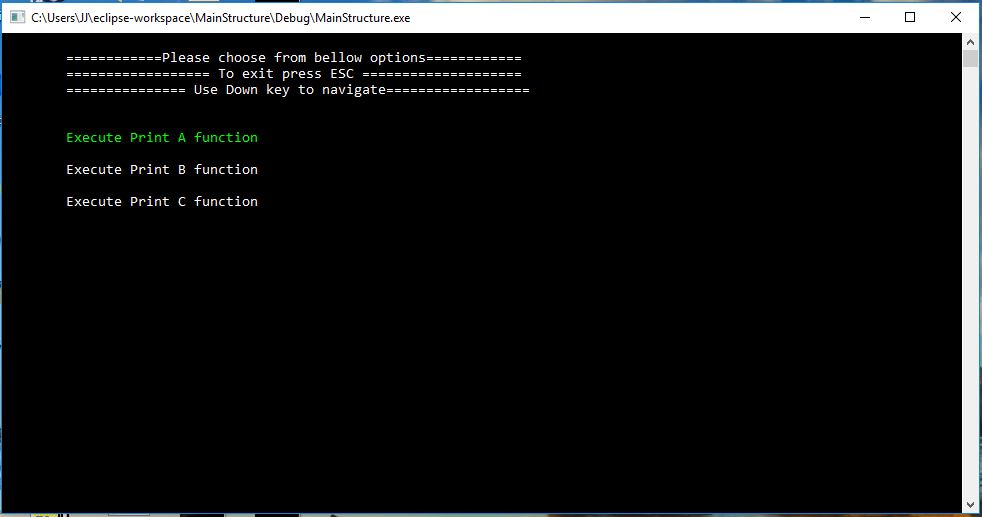
  
Illustration 2: Sub menu#1

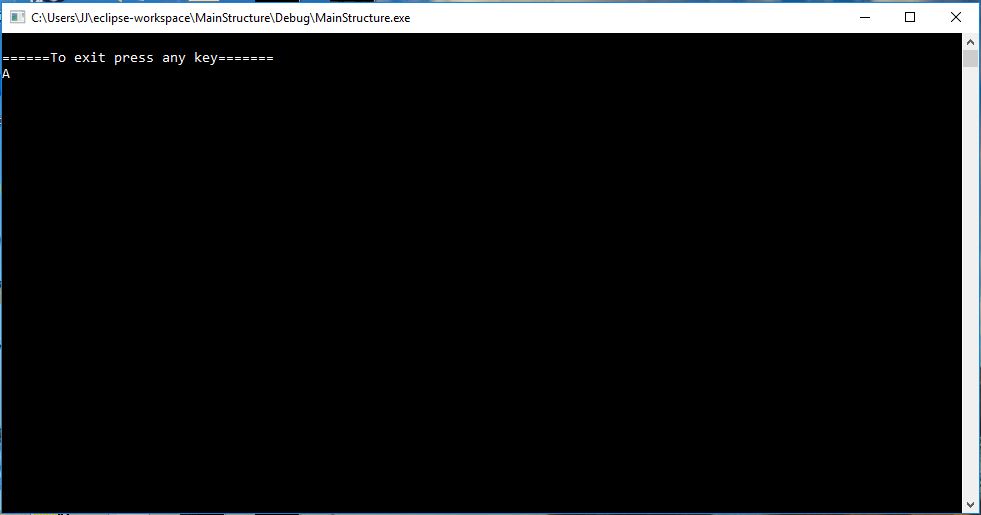
  
Illustration 3: Input and out put for sub problem 1.1

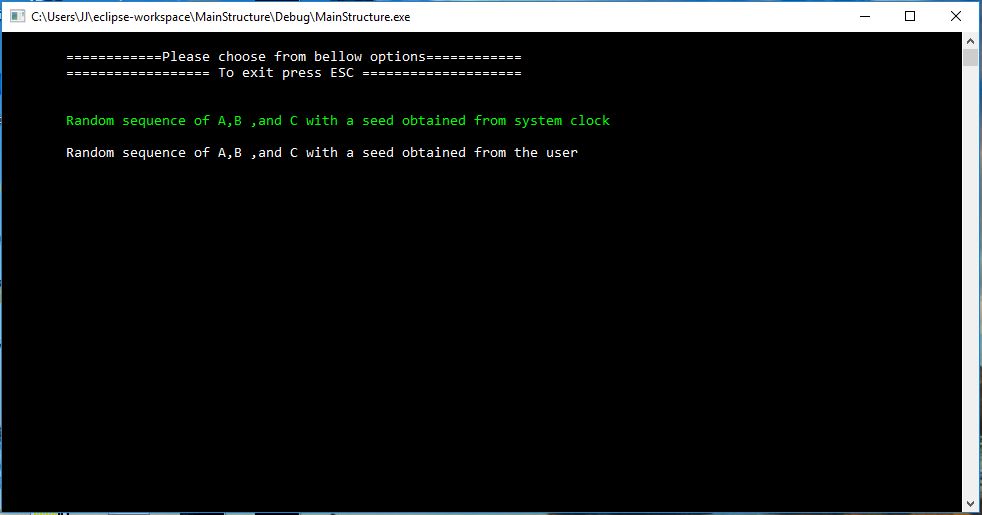
  
Illustration 4: Input and out put for sub problem 1.2

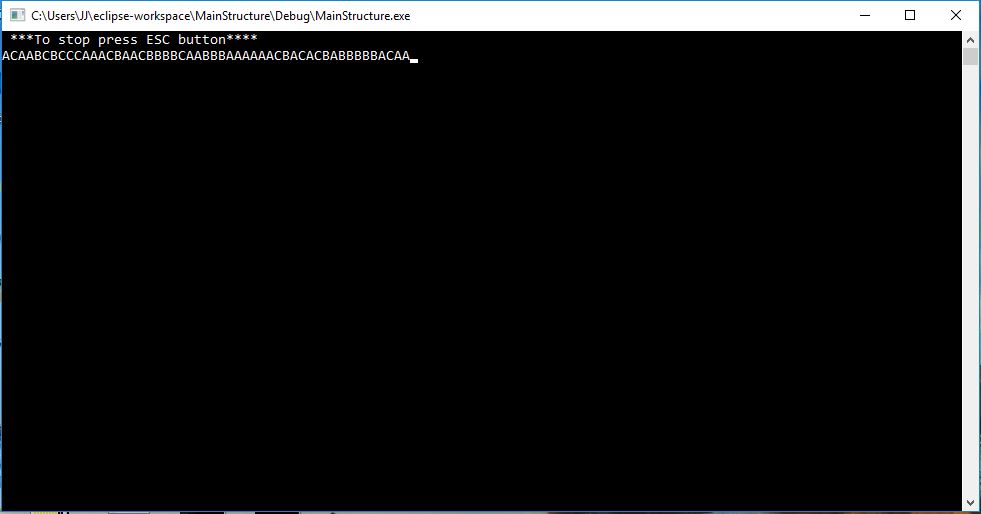
  
Illustration 5: Input and out put for sub problem 1.3

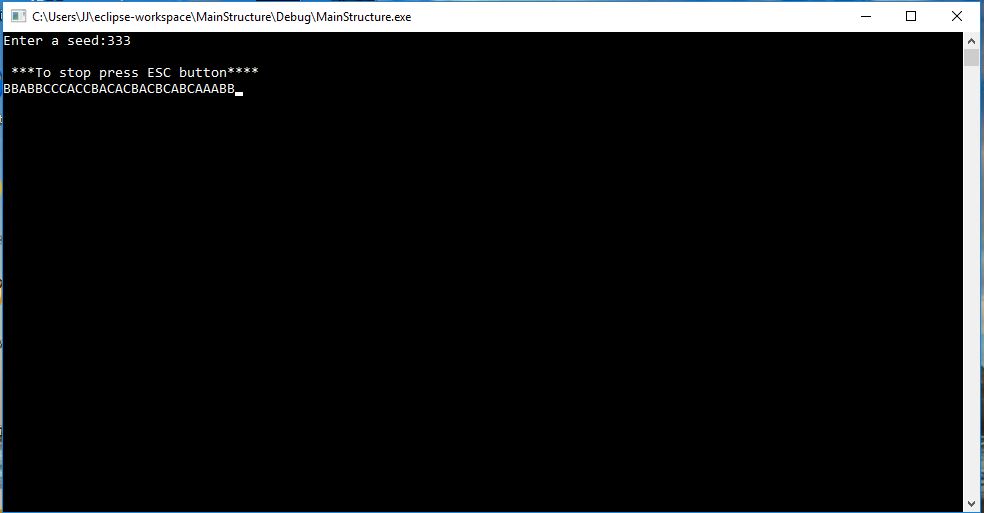
  
Illustration 6: Input and out put for sub problem 1.4

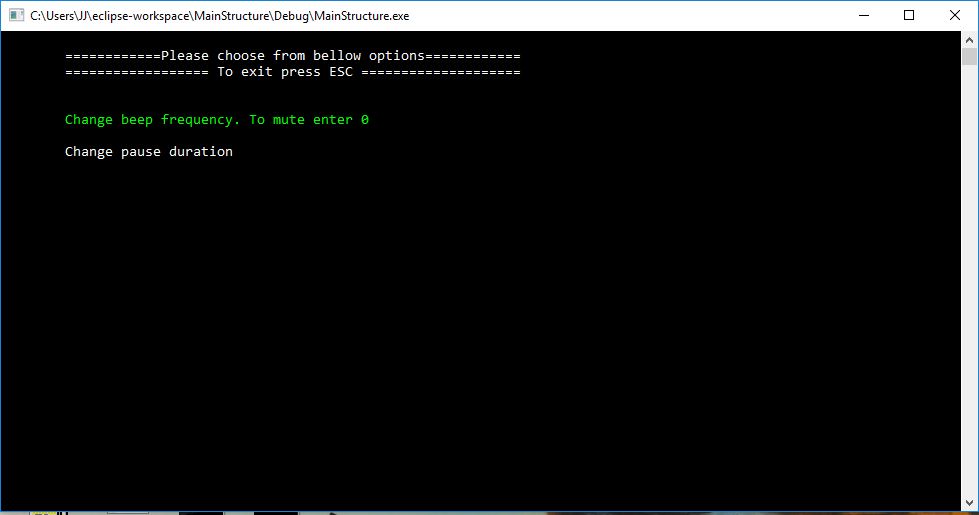
  
Illustration 7: Sub menu #2

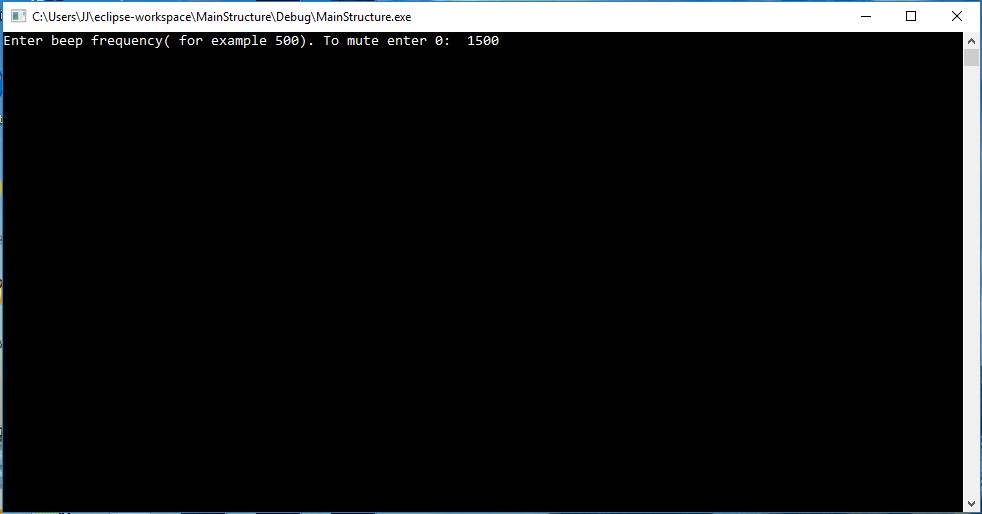
  
Illustration 8: Out put for sub problem 2

  
Illustration 9: Sub problem #3 menu

  
Illustration 10: Input and out put for sub problem 3 part 1

  
Illustration 11: Input and out put for sub problem 3 part 2

  
Illustration 12: The option menu

  
Illustration 13: Option menu, changing the beep specification

Appendix A codes: