T.C.
DOKUZ EYLUL UNIVERSTY

FACULTY OF ENGINEERING

DEPARTMENT OF COMPUTER ENGINEERING

2023 - 2024 FALL SEMESTER

CME 3205 OPERATING SYSTEMS

ASSIGNMENT 1 ARRAY ADDITION SERVER

DUE DATE 23:55 – 27.12.2023

In this assignment you are asked to create a simple array addition server, which will take two series of numbers from a port connection using telnet program (telnet console command that has been used in lab sessions), perform an addition operation on them and output the result to original telnet connection.

The integers you accept and work on should be at most 3 digits long (from 0 to 999). If the input contains number(s) larger than 999 and smaller than 0, you should just return an error. The number of integers should also be equal for both input integer arrays. If they are not equal, this should also return an error.

Your program should be written in C programming language and it should work on a Linux computer. You should create a simple virtual Linux operating system inside VirtualBox software (the same setup we use in our lab sessions) for development and testing of your assignment.

A sample port/socket communication between your assignment and telnet program is given below with extra information and comments.

`telnet localhost 60000`
With this command, you will be connected to port 60000 of your localhost (your own computer). Assuming this is the port that your assignment is currently working on, you should receive the following message.

`Hello, this is Array Addition Server!`

`Please enter the first array for addition:`
`105 449 445 842 292 655 959 6 404 149`
To simplify input, we will just use single spaces for integer separation. Commas or parenthesis's are also not necessary. As you can see, there are 10 integer inputs for the first array.

`Please enter the second array for addition:`
`999 601 78 502 156 482 805 670 834 27`
Just like before, we input 10 different integers for this operation, different number of integers should raise an error.

`The result of array addition are given below:`
`1 105 50 524 344 449 138 764 677 238 176`
As you can see, if the sum of two numbers are greater than 999, only the first 3 digits are stored and written in its original location and a carry is added to the integer at left. This results on an additional integer generated with carry at the leftmost location of array.

`Thank you for Array Addition Server! Good Bye!`
With this message, the connection is closed by the server and communication ends.

The above example shows how a communication between your server and telnet command should be handled. In addition to this you should also send the following error messages, if the required conditions happens.

If the number of integers that are send to server for first and second array are different:
ERROR: The number of integers are different for both arrays. You must send equal number of integers for both arrays!

If the input arrays contain non-integer characters:
ERROR: The inputted integer array contains non-integer characters. You must input only integers and empty spaces to separate inputted integers!

If there is an error that is not covered here, you can write your own error message with similar style to given error messages above. After sending an error message, you should close the connection and wait for new telnet connection to be made.

You are also required to use following global variables to make your code more understandable and standardized.

int DATA_SIZE = 100;
The maximum number of input string size and integer array size your program should work for.

int PORT_NUMBER = 60000;
The port number that will be used by your program.

```
char INPUT_STRING [DATA_SIZE];
int  FIRST_ARRAY  [DATA_SIZE];
int  SECOND_ARRAY [DATA_SIZE];
int  CARRY_ARRAY  [DATA_SIZE];
int  RESULT_ARRAY [DATA_SIZE];
```
These arrays are used to store inputted string from telnet and integer arrays that you have created from this input string for first and second array. The carry array are used to store carries from addition operation to be added to the next operation.

pthread_t THREAD_ARRAY [DATA_SIZE];
This array is used to create and store threads that you will create to calculate every step of the addition operation.

These are the basic variables and their names you should use on your program. A standard pseudo code algorithm/operation of your program is given below.

First your server starts operation and waits for telnet connection.

When a telnet connection is received, the communication steps that has been given above are performed.

Your server receives the first integer array as a string and stores it in "INPUT_STRING" array.

Your server transform "INPUT_STRING" array into "FIRST_ARRAY" using string operations of C language.

Your server receives the second integer array as a string and stores it in "INPUT_STRING" array.

Your server transform "INPUT_STRING" array into "SECOND_ARRAY" using string operations of C language.

After both arrays are received and transformed, your server creates threads for every index of integer array that contains an integer (You need to check how many integers you received from telnet connection while you are doing transformation from string to integer array).

The threads created by your server, performs addition operation between elements of "FIRST_ARRAY" and "SECOND_ARRAY" and stores the result in "RESULT_ARRAY". Every thread is assigned to a single address/index of integer arrays. If the operation results in a carry (value larger than 999), thread assigns a carry (1) to the corresponding address in "CARRY_ARRAY".

After every thread is finished operation, your server adds the calculated carries to their correct address in "RESULT_ARRAY" and finishes addition operation.

Your server transforms "RESULT_ARRAY" to a string (in "INPUT_STRING" char array if you wish) and sends the result to telnet connection.

Your server closes connection to this particular telnet connection and waits for new telnet connection to repeat the steps that has been shown here.

This is the main algorithm/operation that your assignment should follow. You should consider this array of numbers as a one very long number that is divided to 3 digit parts (similar to how we write numbers).

In addition, you should only consider positive numbers for this program, if a negative number is given, it should return an error. The carry we mentioned before is the same carry in addition operation. You output a carry if addition of two numbers in any step/address is more than 999.

You should explain in your comments and on your code control how did you placed the input numbers you received to an integer array, was the LSB (least significant bit, smallest digit) located in first address or was it MSB (most significant bit, highest digit) located in first address (array[0] in other words)? You are required to explain it on your code control and explain it in your code comments as well. An example of array placement of previously given number with different ordering is given below.

```
  0   1   2   3   4   5   6   7   8   9 (MSB at adress 0)
105 449 445 842 292 655 959   6 404 149 (Example number)
  9   8   7   6   5   4   3   2   1   0 (LSB at adress 0)
```

You should also check if it is possible for you to detect if the input string was over limit (more than 100 characters assuming the default value, you can lower it to test it) or not. If you can detect this, you should return an error explaining to problem to the telnet connection. You should also explain it on your comments and code control and show it. If you cannot detect this, again show and explain it on your comments and code control.

You should also consider a possible carry problem at the end of addition operations. Assume that you have a step/address that is 999 and there is a carry from lower addition step. As you can see, you need to change the integer value here to 0 and create a carry for next step of addition. This should be done by main server thread after all addition operations have been completed by child threads.

As it is stated above before, any integer that is larger than 999 should result in an error. You should inform the telnet connection of this error and close connection.

Your program should also be able to process multiple spaces if it is used. For your program “999 999” and “999 999” should be the same input. To simplify your program, you should not process tabs (“\t” character in string operations) and if you encounter tabs, just return an error to telnet connection explaining what caused the error.

You are free to use functions that take a string that contain a single number (“999”) and transform it to corresponding integer value (999). Standard C language libraries contains functions such as this, however, the initial splitting of input string into multiple integer strings should be done by your program. You are not allowed to use a function or library that will take a string that contains multiple integers (“100 101 102”) and return an integer array with given integer values ({100, 101, 102}) or any similar operations.

UPLOAD REQUIREMENTS:

You are required to upload the C language code file you have written to the SAKAI. You should compile and test it to make sure it works before upload. You can use a IDE during development but your code must work correctly using console complication and execution commands (using “gcc” and “./”). If we cannot correctly compile and execute your code in our computers, your grade will be significantly reduced due to not being able to see results to evaluate. You do not need to upload a complied version of your code, just your C language code, because we cannot prove if it is a original compilation or not. For this reason, uploading a complied file is not necessary nor it is requested.

The file you are required to upload are given below with an explanation and an example. You are free to do this assignment as a group of up to three (3) students or individually. Please do not use empty spaces or Turkish language characters in your file name.

For 3 Student Groups:
(STUDENT_NUMBER_1)_(STUDENT_NAME_1)_(STUDENT_NUMBER_2)_(STUDENT_NAME_2)_(STUDENT_NUMBER_3)_(STUDENT_NAME_3).c
Example: 2020510123_fatih_dicle_2020510124_yunus_dogan_2020510125_ali_veli.c
(Source code you have written in C language)

For 2 Student Groups:
(STUDENT_NUMBER_1)_(STUDENT_NAME_1)_(STUDENT_NUMBER_2)_(STUDENT_NAME_2).c
Example: 2022510123_fatih_dicle_2022510124_yunus_dogan.c
(Source code you have written in C language)

For Individual Students:
(STUDENT_NUMBER)_(STUDENT_NAME).c
Example: 2022510123_fatih_dicle.c
(Source code you have written in C language)

The criteria that your program will be graded on are given below.

File and code criteria:
- ***Correct naming of upload file.***
- ***Correct English variable names.***
- ***Correct English comments.***
- ***Correct code quality, indentation and readability.***
- ***Using required variables with correct names.***
- ***Using correct port number.***
- ***Explanation of placement of input integers in your arrays on your comments and code control.***

Error detection criteria:
- ***Correct error detection and output for different number of integers.***
- ***Correct error detection and output if any of the input numbers are larger than 999.***
- ***Correct error detection and output for non integer and space characters (including tabs and negative symbol).***
- ***Correct error detection and output if the input string is larger than supported maximum (given as 100 above).***
- ***Explanation of error detection and output on your comments and code control.***

Execution criteria:
- ***Sending required information to telnet.***
- ***Receiving input array strings from telnet.***
- ***Correct input transformation from string to integer.***
- ***Correct processing multiple spaces instead of single spaces if they are present in input string.***
- ***Using threads for every address of addition operation.***
- ***Correct addition operation and carry generation.***
- ***Correct carry calculation and addition to result.***
- ***Correct handling of possible 999 result and previous carry situation in final calculation step.***
- ***Correct output transformation from integer to string.***
- ***Sending required information and result to telnet.***
- ***Closing the connection after finished operation.***

Late or no submissions will be graded zero. In a rare case of system or upload failure, send your code file to research assistants as an email. This does not guarantee that your code will be accepted or graded but it will be a record of your problem and your code. For this reason, you should finish your code one or two days before your deadline and make an upload, instead of leaving it to last day and hour. If you want to improve upon it before deadline and a new upload, just delete the original upload and make a new one and add a date time string with format "YYYYMMDD_HHMMSS" to the end of your file. An example is given below (you can use the same format for group uploads as well).

For Individual Students:
(STUDENT_NUMBER)_(STUDENT_NAME)_YYYYMMDD_HHMMSS.c
Example: 2022510123_fatih_dicle_20231227_235123.c
(Source code you have written in C language)

You should try to make an upload of your code (even if it is not yet finished) one or two days before deadline to have proof of your code in the system. If you improve upon

Your uploads will be analyzed for cheating and plagiarism and if it is detected, your assignment will be graded as zero. You can search and learn from online sources for this assignment but write your own source code instead of copy pasting. If you do not, that will be considered as plagiarism.

The student groups will be asked and assigned on your lab sessions, preferably on your lab sessions at dates 30.11.2023 and 07.12.2023. You are required to form your groups and inform the research assistant in your lab sessions at given dates.

The date and time of your code control will also be asked and assigned on your lab sessions. The details will be announced either in your lab sessions or on announcement page in Sakai page of this class. The code control will be performed at the last two lab sessions (lab sessions at dates 28.12.2023 and 04.01.2024). If there is a change with the given dates, it will be announced either in your lab sessions or on announcement page in Sakai page of this class.

You are required make an upload at given deadline and If you do not participate on code control on requested lab sessions (even if you made an upload), your assignment will be graded as zero.

If you have any questions or problems regarding this lab paper, you can ask about it in our lab sessions or in Sakai class forums.

However, please do not send emails because that would lead to asking the same questions over and over again.

GOOD LUCK TO YOU ALL!