[CS200]-STT II **Fall 2020-21**

Assignment: IV

Name: Shubham Gupta RollNo: 11941140 Email: shubhamgupta@iitbhilai.ac.in

Collaborators Names: Aayush Deshmukh(11940010), Santaz Sahiti(11940230)

Question 1,
a) The value of the variables are calculated in the gdb log file.

The following table of the values of variables at the marked line numbers in the CPP Program are:

Line Number	X	result	answer	y
07	0	0	NDSC	NDSC
10	10	0	NDSC	NDSC
12	10	35	NDSC	NDSC
17	0	\0	0	10
21	20	\0	15	10
23	20	\0	35	10

Note: NDSC written in the table above signifies ”Not Defined in System Context”
Note: \0 written in the table above signifies ”String \000 <repeat 113 times>”.

The gdb commands used are as follows:

-> break 7(likewise for 10, 12, 17, 21, 23) .....Set the break-points
-> run .....Start running the code with the break-points
-> print X(likewise for result, answer, y) .....Obtain values at desired break-point
-> continue .....Proceed to the next break-point
(Repeat the above 2 lines for all the 6 breakpoints and obtain the desired value)
-> info break .....Verify that all breakpoints are covered with information
-> quit .....Exit the gdb

b) The table in Part (a) of the question, the value of result = 0 at line 10.

On the running of the program, the main function runs and on the running of the main (line 5-13) all the integer variables are initialised to 0.

In the gdb log file, we have run the file 10 times with a break-point at 10 and everytime printing result is giving 0.

On execution, the main scope of the function initialises all integers to 0 unless pre-assigned by the programmer. Since result is not pre-assigned so it initialised to 0.

Every time on compilation, the same principle applies and so the value of ”result” variable is always set to 0.

The gdb commands used are as follows:

-> break 10 .....Set the break-point
-> run .....Start running the code with the break-points
-> print result .....Obtain value at desired break-point
-> continue .....Proceed to the next break-point
(Repeat the above 3 lines for 10 times to check the value of result)
-> quit .....Exit the gdb

c) The table in Part (a) of the question, the value of X(line 10) = 10 and X(line 17) = 0.

In line 17, the code is inside the ’func’ function which is a differrent module in the program.
In any part of a module of the code, CPP initialises new integer variables to zero.
On line 17, the value of X has not been used nor initialised in the ’func’ function.

However, X has been initialised as a global variable(line no 4, const int X = 5).
Since, X is a global variable, so it is defined in the system context as the ’func’ function.
However, on entering the new function from the global scope, all integer variables are re-initialised to 0 unless pre-assigned by the programmer. In this case, the global variable is re-initialised in the ’func’ function as 0 as it is not pre-assigned by the programmer.

In line 10, of the ’main’ function, we are in a differrent module of the code from the global scope and hence X will be re-initialised to 0 unless pre-assigned by the programmer. However, in line 9 of the ’main’ function(int X = 10), X has been pre-assigned to 10 by the programmer and so X is printed as 10 as expected.

The only contrast between the values is due to the re-initialisation of the values in the modules of the CPP Program. In line 17 as X is not pre-assigned by the programmer so X = 0 but in the line 10, X is pre-assigned due to which X = 10.

The gdb commands used are as follows:

-> break 10 (likewise for 17).....Set the break-points
-> run .....Start running the code with the break-points
-> print X .....Obtain values at desired break-point
-> continue .....Proceed to the next break-point
(Repeat the above 2 lines for both the breakpoints and obtain the desired value)
-> quit .....Exit the gdb.