How to Run the Program

[Nada Badawy – Eman Assem]

MIPS coding: sprintf

The Design

Our program consists mainly of two parts, the parsing and the sprintf which contains the whole conversion.

**The parsing:**

Before parsing the string, the program asks the user to enter a string with the same format for the printf function. Then the parsing part starts to run. At the first it loops the whole string and go out from the loop if it finds a “;” which indicates the end of the input string. While parsing the input string the parsing part itself consists of two loops; the format loop and the variable loop. The main function for the format loop is that it stores the required format from the user which will be indicated by % sign. It stores it in the formatting string which is being declared in the data part as space. Moreover, the main function for the variable loop is to extract the variables from the input string and store them as arguments two of them will be added in a2 and a3 and the rest three will be stored in the stack to be retrieved later. After finishing and set all arguments of sprinf function the program calls it.

**Spritf:**

At first the function stores the variables a2, a3 and from the stack to the local variables in the function. Then take the requirement format and put it in array. They are total of 8 operations; signed integer and output in decimal, unsigned integer and output in decimal, unsigned integer and output in binary, unsigned integer and output in lowercase hexadecimal, unsigned integer and output in uppercase hexadecimal, unsigned integer and output in octal, treat the low byte of the argument word as a character and copy it to the output and treat the argument as a pointer to a null-terminated string to be copied to the output. Each one of these functions uses a different version of itoa function based on the functionality. After calling the corresponding itoa function, the subfunction then copies the result of the itoa function to the output buffer and returns the updated output string.

**How to Run the Program**

The program to be able to run in appropriate way must have a specific format. For instance, the required format should be written in a quotation mark “” and must be preceded by % sign to be recognized as a format type. Moreover, the variable should follow them. The maximum number for this program is five. So, the user has to put the variable and put comma between them.

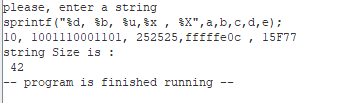
* Step1: the program will ask the user to enter a sting with the format explained before.
* Step2: the program will output the output string according to the user input

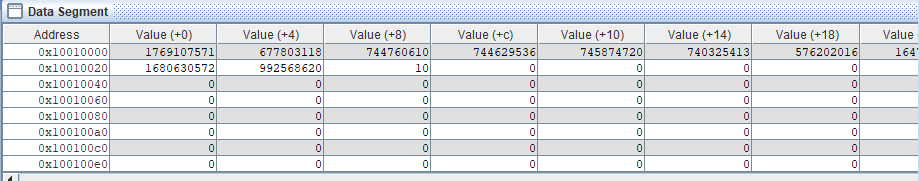
Hint: the values of the variables are stored in the program itself as register from $t5 to $t9.

**Test cases:**

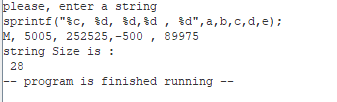
Case1:

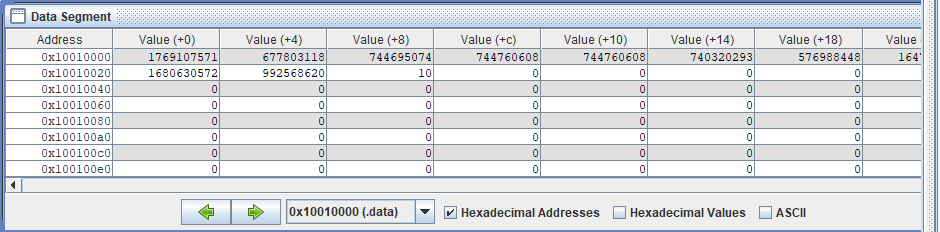
with a=10 , b= 5005, c= 252525, d= -500, e= 89975

test d, b, u, x and X

the memory:

case2:



In memory:

**Challenges**

* We faced a challenge when storing the argument in a2, a3 and the stack. At first, they were overwriting each other. In order to solve this, we declared a register to work as a counter of the arguments to help in putting the arguments in its wright place.
* Second challenge was implementing %s which treats the argument as a pointer to a null-terminated string to be copied to the output.

**Limitation**

* The program is limited for 5 arguments and they are pre-stored in the program which requires for each time to be changed to change them in the program itself
* The fixed format for the input string limits the program and also any change in this format will lead to the program to not work.