International Journal of Research in Advanced Engineering and Technology
ISSN: 2455-0876; Impact Factor: RJIF 5.44
Received: 03-04-2019; Accepted: 05-05-2019
www.newengineeringjournal.in
Volume 5; Issue 3; July 2019; Page No. 10-12

Code Miniaturization: A means for enhancing user's applications and system efficiency

Barida Baah[1], Shedrack Mmeah[2], Abasiama G Akpan[3]
[1] Department of Computer Science, Ebonyi State University, Abakaliki, Nigeria
[2] Department of Computer Science, Ken Saro Wiwa Polytechnic, Bori, Rivers, Nigeria
[3] Department of Computer Science, Evangel University, Akaeze, Nigeria

Abstract
The efficiency of every application's that is developed is dependent on the magnitude of the code on the system which is the main reason for this research work. It aim is to create awareness and also to make source codes of an application to be small in size, so as for the application to able to run faster and also avert most of the problems normally, associated with application that are large in size in terms of code resulting to application inefficiency or system failures. To illustrate this notion of code miniaturization a flowchart is used to show the miniature flow and non miniature flow and a simple miniature source code and non-miniature source code was created in Matlab to sort numbers in ascending order as shown in Table 1. The methodology implore in this research work is the Object Oriented Analysis and Design Method (OOAD). The benefits of miniaturized code is it that it utilizes less memory space, less computational time, makes the mobility of code to very fast and it also improves on the application that is develop and systems efficiency in terms of job loads.

Keywords: code miniaturization, application and efficiency

1. Introduction
The idea of making a source code to small in size should be an important aspect to every programmer who are into software development since it is very useful in terms of improving on the efficiency application program that are been develop for a specific purpose. Whenever a developed application is miniaturized it utilizes less memory space that is to be occupy by the source code and also it makes developed software program to run faster on the system without unnecessarily halting the system during the execution of the sources code. non-miniature source code can be likely to a human being carry a heavy load on his/her head this will certain affect the individual mobility due to the size of the load on his/her head thereby slowing down the movement of such person carrying such load.

2. Literature review
The concept of miniaturization has been applied mainly in the areas of hardware as opined by many researchers. The survey literature is discussed below:
Microchips which are hardware have been designed for a range of chemical and biological analyses based on chromatography, electrophoresis, immunoassay, and nucleic acid target and probe amplification (e.g., PCR and the ligase chain reaction). They have also been very effective for cell isolation and selection by use of microfiltration or electrical fields, and the popular Coulter counter and flow cytometer have been successfully miniaturized into a chip format [1, 2]. Microchip components are also gaining popularity for sample application in electrospray mass spectrometric methods [3].
Manz A *et al*. [4] opined that an important advantage of the microchip approach to analysis is integration of successive steps in an analytical process (e.g., sample preparation, analytical reaction, and detection) on a single microchip or a multilevel microchip to produce a miniaturized total analytical system or a lab-on-a-chip. In this case it only focuses on hardware microchips miniaturized it fails to address the issue of software miniaturized coding. The scope of micro machined components available for incorporation into a lab-on-a-chip is diverse and includes pumps, valves, fluid channels and chambers, thermal control systems, sieves, and filters.
Key to the development of this new type of analyzer has been refinements of microfabrication techniques, development of convenient microchip-user interfaces, and a better understanding of microfluidics. Different types of microfabricated on-chip valves and pumps have been designed, and flow within chips can also be controlled by electrokinesis. An advantage of the latter option is that there are no moving parts, only the electrodes that are used to control flow. A particular concern for microchips
has been the efficiency of mixing within submicroliter chambers and channels. New modeling software allows simulation of flow within microchips, and hence design optimization, before microchip fabrication [5].

Flowchart

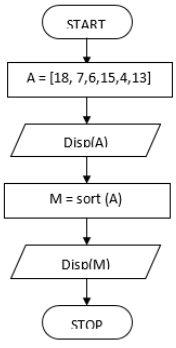

Fig 1: A Miniature flowchart showing use of function to sort numbers in ascending order

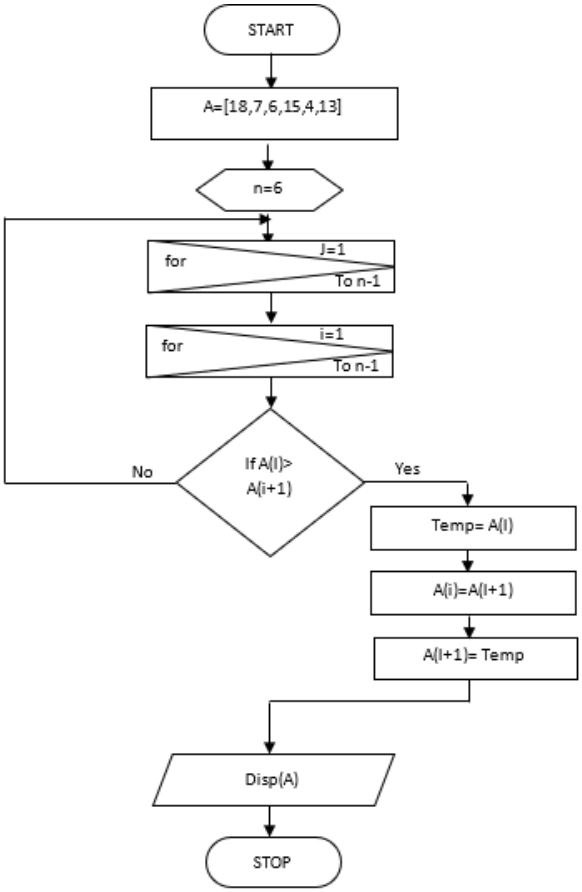

Fig 2: Non-miniature flowchart without use of function to show sorted numbers in ascending order

2.1 Element of Code Miniaturization

These has to do with the element or feature that are present in code miniaturization which is the use of function in programming, most of the programming languages these days have an inbuilt function that is imbedded in it, which make it possible for programmers to developed a very fast codes that will do a lot of work.

2.1.1 Functional Programming

The use of functional programming should be applied by programmers when developing their codes since it is an important aspect when it comes to miniaturizing codes. It assist programmers to achieve their aims when using the available function in that programming language to save time of coding, reduce the size of the code thereby making it very effective and also ensure that program run fasters and avert problems of unnecessary system failure normally encountered while running codes that are also very large in size (ie non-miniature source code).

Table 1: A Miniature and Non miniature sources codes that sort numbers in ascending order

Miniature Source Code	Non-Miniature Source Code

Disp ('Unsorted Numbers')
Disp (")
disp (")
A=[18 7 6 15 4 13];
disp(A)
disp('Sorted Numbers in Ascending Order')
M= sort (A);
Disp (M)
	Disp ('Unsorted Numbers')
	Disp (")
	Disp (")
	A=[18 7 6 15 4 13];
	disp(A)
	n = 6;
	%Module to Perform of Numbers by Bubble sort
	for j = 1:1: n-1
	% comparing the next number and then swap it
	for i = 1: 1: n-1
	if A(i)> A(i+1)
	temp = A(i);
	A(i) = A(i+1);
	A(i+1) = temp;
	end
	end
	end
	% Display the Sorted Numbers in Ascending Order
	disp('Sorted Numbers in Ascending Order')
	disp('------------------------')
	disp(A)

From table 1 above shows the two sample source codes in which one miniature and the other non-miniature source codes of a program that sort numbers in ascending order given 6 numbers that are unsorted, the miniature source code is small in size and will take less memory spaces, less computational time and fast execution time compare to the non-miniature source code which will take more memory space, more computational time and slow in execution and most a time will result to system or application failure.

Conclusion

In conclusion, the used of function to ascertain a miniature source code should be a serious business by programmers since it has to do with the reducibility of the source code which make it possible for programmer(s) to developed a very powerful program that is capable to doing more tasks, take less memory space, less computational time and avert the problems that are normally associated with the program that are non-miniaturized; most often it will result to system or application failure during the program execution.

References

1. Koch M, Witt H, Evans AGR, Brunnschweiler A. Design and fabrication of a micromachined Coulter counter. J Micromechan Microeng. 1999; 9:159-61.
2. Pan LS, Ng TY, Liu GR, Lam KY, Jiang TY. Hydrodynamic focusing for a micromachined flow cytometer. J Fluids Eng Trans ASME. 2001; 123:672-9.
3. Xue Q, Dunayevskiy YM, Foret F, Karger BL. Integrated multichannel microchip electrospray ionization mass spectrometry: analysis of peptides from on-chip tryptic digestion of melittin. Rapid Commun Mass Spectrom 1997; 11:1253-6.
4. Manz A, Graber N, Widmer HM. Miniaturized total chemical analysis systems: a novel concept for chemical sensing. Sens Actuators. 1990; B1:244-8.
5. Coventor web site, 2002, www.coventor.com Accessed.