Université d'Ottawa
Faculté de génie

École de science informatique
et de génie électrique

uOttawa

L'Université canadienne
Canada's university

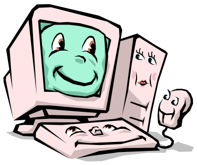
University of Ottawa
Faculty of Engineering

School of Electrical Engineering
and Computer Science

GNG 1106
Fundamentals of Engineering Computation

Lab 6 – Winter 2016

G. Arbez



Lab 6 Review of Functions and Arrays

Programming aspects to get familiarized with:

- Working with Functions.
- Review the use of arrays.

Complete the exercises and show your working program to the TA for extra marks to be used to adjust your midterm exam.

A. Exercise : Simple Functions – Calculating Velocity and Displacement (2 marks)

Statement of the problem

Develop a program that requests from the user the initial velocity at time 0, v_0, and the acceleration, a, of an object travelling in a straight line and a time t (greater than 0). From these values, the program shall calculate the displacement, x, and the velocity, v, of the object after time t.

Background:

The following equations are well known relationships between the constant acceleration a, velocity v, and displacement x of an object that travels in a single direction (x).

$$v = v_0 + at \qquad \text{Equation 1}$$

$$x = v_0 t + \frac{1}{2}at^2 \qquad \text{Equation 2}$$

where a is a constant acceleration in m/s^2,
v is the velocity in m/s,
x is the displacement in m,
v_0 is the initial velocity at time t_0.

Design:

- The program should be developed with 3 functions, **`main`**, **`calculateVelocity`**, and **`calculateDisplacement`**.
- In the main function, prompt the user to obtain the initial velocity, acceleration, and time t. Call the other functions to compute displacement and velocity, and print the results. (Hint: start with the program from **assignment 1** posted with this lab and modify it to add the two functions as described in the next points).
- Create two functions, **`calculateVelocity`** which calculates the velocity at a given time, and the other **`calculateDisplacement`** which calculates velocity.
- Both functions have the same three parameters: the initial velocity, the acceleration, and time.
- **`calculateVelocity`** returns the velocity, v, of the object at time t.
- **`calculateDisplacement`** returns the displacement, x, of the object at time t.

Test Cases

Velocity (m/s) and displacement (meters) after a time t (seconds) can be calculates given an initial velocity v0 (m/s) and constant acceleration (m/s2) with the equations given in Step 2.

The following table provides test cases that can be used for testing the software.

Time t (seconds)	Initial Velocity v0 (m/s)	Acceleration (m/s)	Velocity (m/s)	Displacement x (m)
10	1	0	1.00	10.00
0.5	0	250	125.00	31.25
120	60	1.2	204.00	15,840.00
0	60	1.2	60.00	0.00

Develop your program and show your TA your working program with one of the above test cases.

B. Exercise: Functions with arrays – Calculating Velocity and Displacement (3 marks)

Modify the program from Exercise A to have the function **`calculateVelocity`** and **`calculateDisplacement`** fill in an array of 100 points according to the following design:

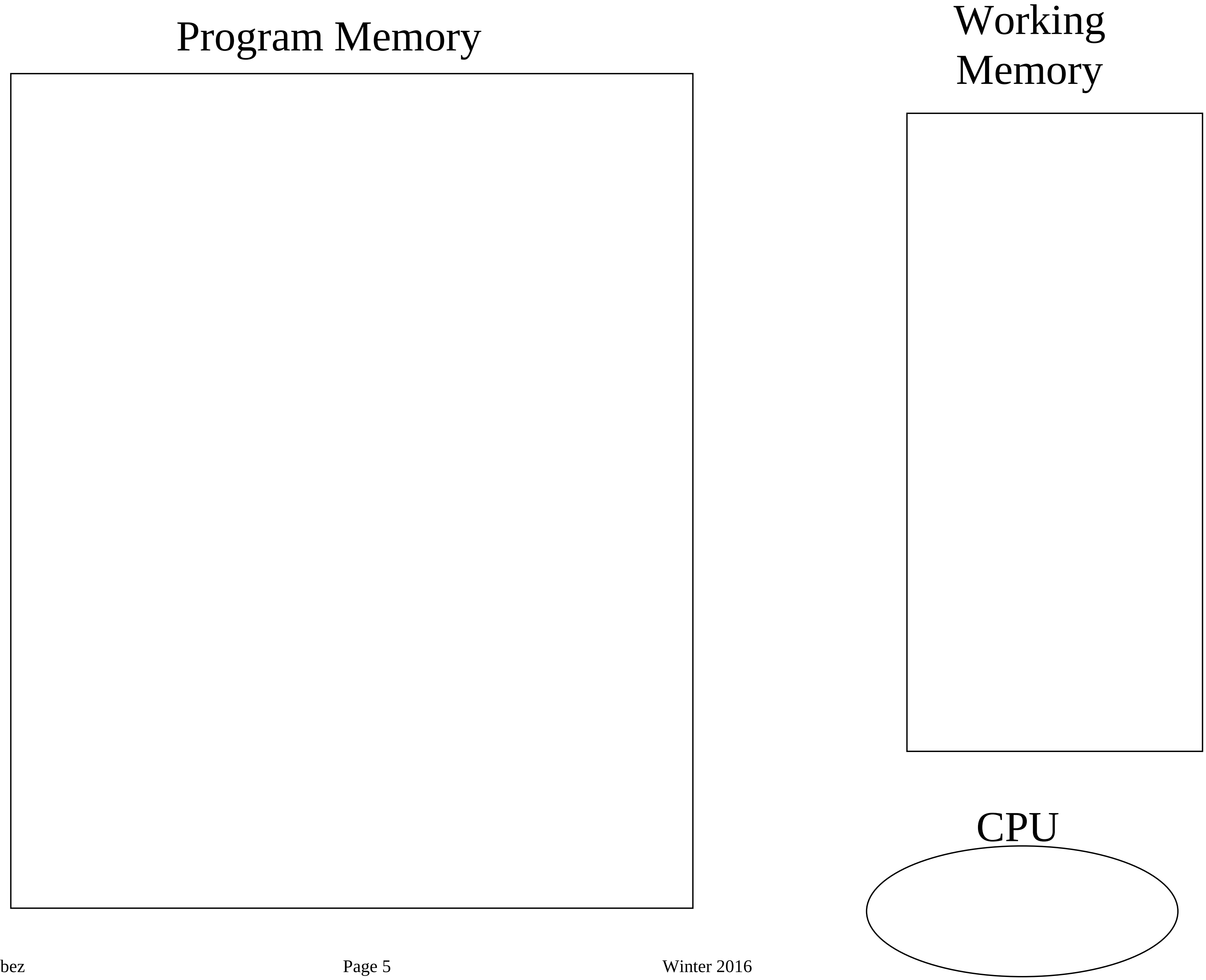
- Define the symbolic constant **`N`** as 100 (number of points to compute).
- In main
  - Prompt the user to obtain the initial velocity, acceleration, and final time **`tfinal`**.
  - Create three arrays, **`tArr`**, **`xArr`**, and **`vArr`** of size **`N`**.
  - Call the **`calculateTime`** function to fill in **`tArr`**.
  - Call **`calculateDisplacement`** function to fill in **`xArr`**.
  - Call **`calculateVelocity`** function to fill in **`vArr`**.
  - Open the file "Results.txt" and save the results (i.e. time, velocity and displacement) in a table form.

- Create another function, **`calculateTime`**, to fill in the time array as follows:
  - Parameters: final time (**`tfinal`**), reference to a time array (**`tarr`**), number of elements in the array (**`n`**)
  - Define an increment, **`inc`**, that is set to **`tfinal/n`**.
  - Traverse the array to fill in time values starting at 0 and incrementing by **`inc`** for each element in the array.
- Modify the **`calculateVelocity`** function as follows:
  - Parameters: initial velocity (**`v0`**), acceleration (**`a`**), reference to a time array (**`tarr`**), reference to a velocity array (**`varr`**), number of elements in the array (**`n`**, note that the symbolic constant **`N`** is not used in by this function.
  - For each time element in the time array, calculate the velocity and store it in the velocity array at the same index position (use a loop to traverse the arrays).
- Modify the **`calculateDisplacement`** function as follows:
  - Parameters: initial velocity (**`v0`**), acceleration (**`a`**), reference to a time array (**`tarr`**), reference to a velocity array (**`xarr`**), number of elements in the array (**`n`**, note that the symbolic constant **`N`** is not used in by this function.
  - For each time element in the time array, calculate the displacement and store it in the displacement array at the same index position (use a loop to traverse the arrays).

Develop your program and show your TA your working program with one of the test cases from Exercise A.

C. Exercise: Functions with arrays – Calculating Velocity and Displacement

If you have time, modify the program from Exercise B to plot the change in velocity and displacement using the plplot libraries. This exercise is optional.



Program Memory

Working Memory

CPU