**CS2100: Computer Organisation**

*Lab sheets uploaded after the respective due time will not be graded.*

**Lab #2: Debugging using GDB II**

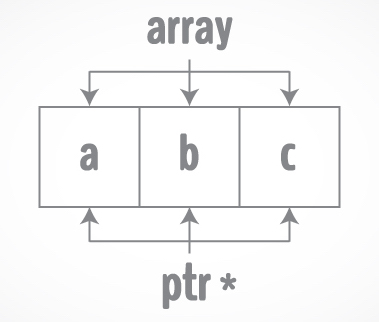
(Week 4: 8 – 12 Sep 2025)

Name: \_\_\_\_\_\_\_\_\_\_Zhu Chongqiao\_\_\_\_\_\_\_\_\_\_\_ Student No.: \_\_\_\_\_\_A0292160L\_\_\_\_\_\_\_

Lab Group: \_\_14\_\_\_

**C Arrays**

Arrays are data structures that store fixed-size sequential collections of elements of the same type. While an array simply stores a collection of data, it is often more useful to think of the collection as a collection of variables of the same type.

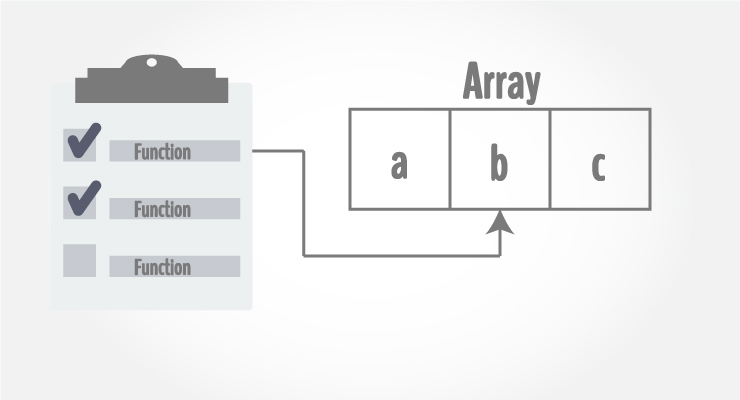
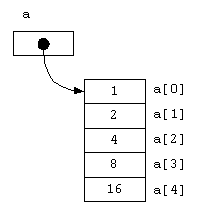


Instead of declaring individual variables, eg. number0, number1... number99, we can declare a single array variable numbers and use numbers[0], numbers[1],…numbers[99] to represent individual variables. A specific element in an array is accessed by an index which starts from 0.

All arrays consist of contiguous memory locations. The lowest address corresponds to the first element and the highest address to the last element.

**C Functions and Arrays**

In C programming, both a single array element or an entire array can be passed to a function. A single value will be passed by value, whereas a whole array is always passed as a reference (think pointer) to the first element of the array. In other words, the array itself is represented by a pointer to the first element of the array.

**** 

**Objective:** You will learn how to use arrays and functions in C.

**Preparation (before the lab):** Please refer to lab#1.

**Procedure**:

1. Download the files **lab2a.c, lab2b.c and lab2c.c** from Canvas.

1. Compile **lab2a.c** with **gcc** using the following command: **gcc –o lab2a lab2a.c**
2. What is the output of the program?



1. Which line in the code should you change to get output **“2”** instead? Show the changed line.  
   **Note:** The output should be related to the **ageArray**. Do not hardcode “2” in your code!

display(ageArray[0]);

1. What is the purpose of the unary operator **sizeof**?   
   What datatype will **sizeof** give the value **“1”** for all architectures?

The purpose of the operator sizeof is outputing the size of the type of operand in bytes.

char

1. Can you get the number of elements in **ageArray**? Write a modified main function below to produce the following output. Show your lab TA the output of the code.

**2**

**Size of the array is 4**

**Note:** The output **“2”** and size of array (i.e., **4**) are related to **ageArray**. Do not hardcode the value “2” and “4” in your code!

The number of elements in ageArray is 3.

int main() {

int ageArray[] = { 2, 15, 4 ,0};

display(ageArray[0]);

printf("Size of the array is %d\n", sizeof(ageArray)/sizeof(ageArray[0]));

return 0;

}

1. Compile **lab2b.c** with **gcc** using the following command: **gcc –o lab2b lab2b.c**
2. Give 2 ways of displaying the stored value of the first element of an array.  
   Give 2 ways of displaying the address value of the first element of an array.

assume the name of the array is a, and the type is int.

printf("%d\n", a[0]); printf("%d\n", \*a);

printf("%p\n", &arr[0]); printf("%p\n", arr);

1. Can you define the function **hexToDecimal(char hex[], size\_t size)** in **lab2b.c**, using pointers to traverse the array?   
   Write your function below and show your labTA the output.

**Note:** You are not allowed to use **strtoul**, **strtol**, or other functions from **stdlib.h**.

*Hint: Reading from the back of array is easier. Furthermore, you are already given the function* **hexVal(char hex)***to simplify your work.*

int hexToDecimal(char hex[], size\_t size) {

int sum = 0;

int multiplixer = 1;

for(char \*p = hex + size - 1; p >= hex; p --) {

sum += hexVal(\*p) \* multiplixer;

multiplixer \*= 16;

}

return sum;

}

1. Why do we pass the size of the array to the **hexToDecimal** function in lab2b.c? Can we calculate the size of the array inside the function?

To get the correct position to move the pointer to the last char of hex. No we can’t. Since C can only pass a pointer to indicate the array, inside the function we can’t get the size of this array.

1. What is the format specifier to print a variable of datatype **size\_t**?

%zu

1. Compile **lab2c.c** using the following command: **gcc –o lab2c –DTEST0 lab2c.c.**What does the option **–DTEST0** do? Hint: read the man page of gcc, i.e. issue the command: **man gcc**.

The -DTEST0 option in gcc defines a preprocessor macro named TEST0.

1. Execute **lab2c** and report what happened. Explain how the output was obtained.

Output:Rhis is a test! In the gcc command -DTEST0 , TEST0 is defined , so the program excecutes the TEST0 is defined branch, replaces the first character of str\_as\_array into’R’ an then prints it

1. Now recompile **lab2c.c** with: **gcc –o lab2c –DTEST1 lab2c.c**. Execute **lab2c** and report what happened. Explain how the output was obtained.

Segmentation fault (core dumped) . In the gcc command -DTEST1 , TEST1 is defined , so the program excecutes the TEST1 is defined branch. Since strings in C are stored in read-only memory, Segmentation fault (core dumped) is output.

1. Now recompile **lab2c.c** with: **gcc –o lab2c –DTEST2 lab2c.c**. Execute **lab2c** and report what happened. Explain how the output was obtained.

Output: his is a test! In the gcc command -DTEST2 , TEST2 is defined , so the program excecutes the TEST2 is defined branch. Then str\_as\_pointer++ moves the pointer 1 byte, which equals skipping the first char of the string.

1. Now recompile **lab2c.c** with: **gcc –o lab2c –DTEST3 lab2c.c**. Report what happened. Explain why.

` lab2c.c: In function ‘main’:

lab2c.c:17:17: error: lvalue required as increment operand

17 | str\_as\_array++;

Got this error. This is because in C the name of arrays are constant hence cannot do addition to the array name str\_as\_array

**Marking Scheme: Report – 16 marks; correct output – 4 marks; Total: 20 marks.**