Remember to bring this along to your lab. Prepare your report before attending the lab!

CS2100: Computer Organisation
Lab #2: Debugging using GDB II
(Week 4: 5 – 9 Feb 2024)

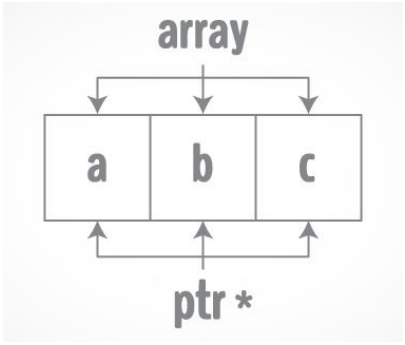
[This document is available on Canvas and course website https://www.comp.nus.edu.sg/~cs2100]

Name: Jonathan Loh Junhao Student No.: A0269683J

Lab Group: B09

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.

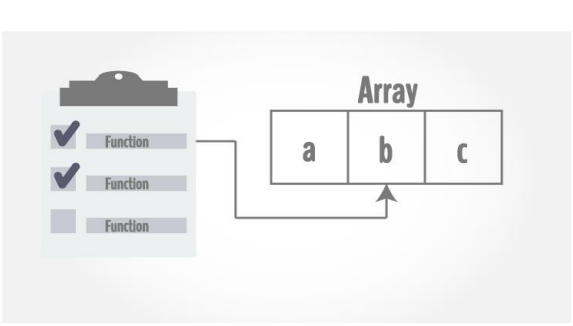


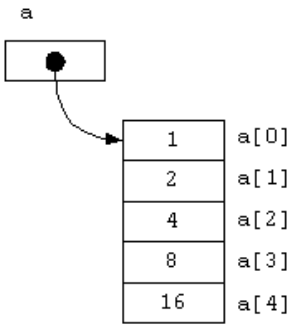
Instead of declaring individual variables, eg. `number0`, `number1`... `number99`, we can declare a single array variable `numbers` and use `numbers[0]`, `numbers[1]`,…`numbers[99]` to represent individual variables. A specific element in an array is accessed by an index which starts from 0.

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 and lab2b.c** from Canvas or CS2100 website "Labs page": https://www.comp.nus.edu.sg/~cs2100/3_ca/labs.html

2. Compile **lab2a.c** with **gcc** using the following command: **`gcc -o lab2a lab2a.c`**

3. What is the output of the program?

   4

4. 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!

   Line 7: display(ageArray[0]);

5. What is the purpose of the unary operator **`sizeof`**?
   What datatype will **`sizeof`** give the value **"1"** for all architectures?

   Returns the size of a variable/data type in bytes. Calculates memory occupied at compile time.

   char

6. 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!

```
int main()
{
  int ageArray[] = {2, 15, 4, 5};
  display(ageArray[0]);

  size_t len = sizeof(ageArray) / sizeof(ageArray[0]);
  printf("Size of the array is %zu\n", len);
  return 0;
}
```

7. Compile **lab2b.c** with **gcc** using the following command: **`gcc -o lab2b lab2b.c`**

8. 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.

```
printf("%d\n", ageArray[0]);
printf("%d\n", *ageArray);
printf("%p\n", ageArray);
printf("%p\n", &ageArray[0]);
```

9. 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 multiplier = 1;
  int result = 0;
  for (int i = size - 1; i >= 0; i--)
  {
    result += multiplier * hexVal(hex[i]);
    multiplier *= 16;
  }
  return result;
}
```

10. 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?

Calling sizeof array will only calculate the size of the array pointer so we need to pass in the size

No. We can only pass a pointer to the array so we need to pre-calculate the size before calling the function

11. What is the format specifier to print a variable of datatype **`size_t`**?

```
%zu
```

Marking Scheme: Report – 11 marks; correct output – 4 marks; Total: 15 marks.