Pointers and Arrays

ESC108A Elements of Computer Science and Engineering

B. Tech. 2017

Course Leaders:

Roopa G.

Ami Rai E.

Chaitra S.

Objectives

- At the end of this lecture, student will be able to
  - apply the concept of pointers and arrays in C programming language

Contents

- Arrays and functions
- Pointers and arrays
- Parameter passing mechanism

Arrays and Functions

- Name of array is constant storing the address of first element

- Function prototype

```
void myFunction(int [], int);
```

- Function definition

```
void myFunction(int myArray[], int myArraySize){
    …
}
```

Arrays and Functions contd.

- C automatically passes arrays to functions by reference

- Passing arrays
  - Specify array name without brackets

    ```
    int myArray[32];
    myFunction(myArray,32);
    ```

    - Array size usually passed to function, unlike char array no special terminator
- Passing array elements
  - Subscripted name in function call

    ```
    myFunction(myArray[10]);
    ```

Pointers and Arrays

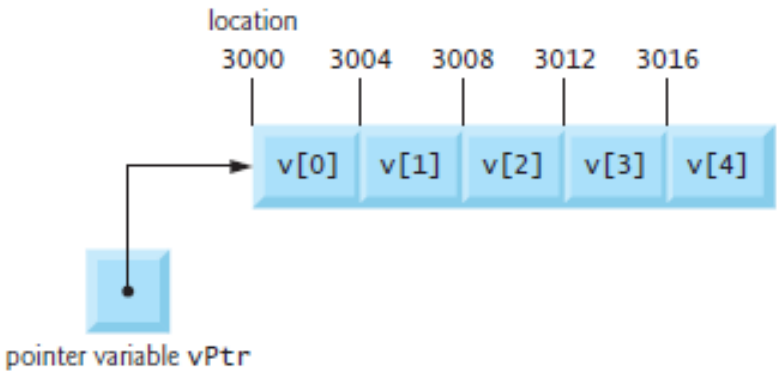
- The address of the first element of the array can be written as &array[0] or array
- Address of the second element can be written as &array[1]
- Generally, address of i^{th} elements is &array[i-1] or (array+(i-1))

- The value at the address [(array+i)] is referenced by *(array+i) which is equivalent to array[i]
  - *(p+5) equivalent to p[5]

Pointers and Arrays contd.

- Variable vPtr can be initialized to point to array

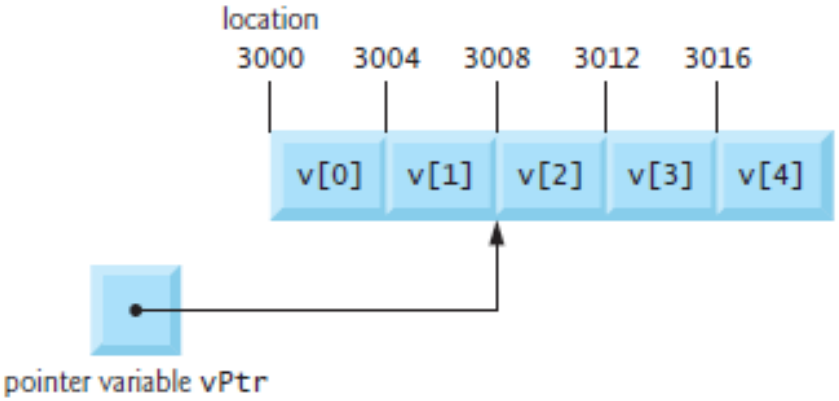
vPtr = v; or vPtr = &v[0];



vPtr += 2;

would produce 3008 (3000 + 2 * 4), assuming an integer is stored in 4 bytes of memory

What is vPtr -= **4;**



Pointers to Arrays

- A pointer variable can be used to access the elements of an array of the same type

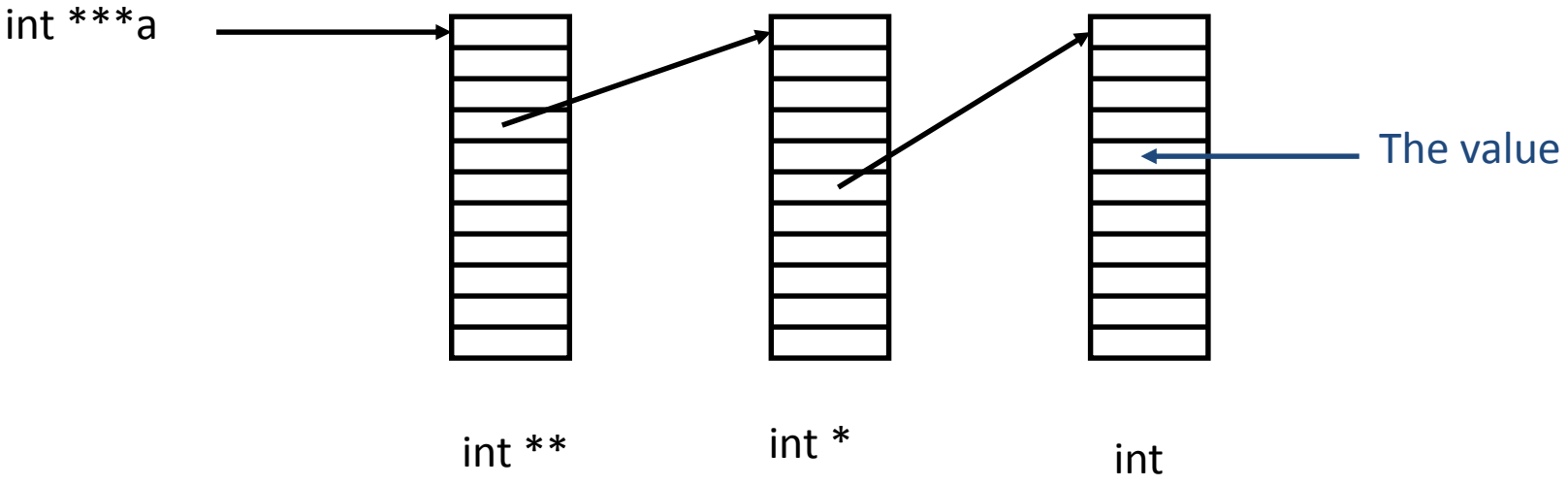
```
int gradeList[8] = {92,85,75,88,79,54,34,96};
int *myGrades = gradeList;
printf("%d", gradeList[1]);
printf("%d", *myGrades);
printf("%d", *(myGrades + 2));
printf("%d", myGrades[3]);
```

Multidimensional Arrays

- Use arrays of pointers for variable-sized multidimensional arrays
- You need to allocate space for and initialize the arrays of pointers

  int ***a;

- a[3][5][4] expands to *(*(*(a+3)+5)+4)



Pointers and Strings

- String Declaration
- A variable of type **char ***
  - char *colPtr = "blue"; //creates pointer variable colorPtr that points to the string "blue" somewhere in memory

- A string is accessed via a pointer to the first character in the string
- The value of a string is the address of its first character
- A string is a pointer
  - A pointer to the string's first character

Pointers and Strings - Example

- Using pointers to access the array elements

```
char name[]="MSRUAS";
char *ptr;
ptr=name;
while(*ptr!='\0'){
    printf("%c ",*ptr);
    ptr++;
}
```

Arrays of Pointers

- Arrays may contain pointers
- A common use of an array of pointers is to form an array of strings

  const char *suit[4] = { "Hearts", "Diamonds", "Clubs", "Spades" };

  – The char * portion of the declaration indicates that each element of array suit is of type “pointer to char.”
  – Qualifier const indicates that the strings pointed to by each element pointer will not be modified

Pointers to Functions

- A function, like a variable has an address location in the memory
- It is possible to declare a pointer to a function, which can then be used as an argument in another function

- A pointer to a function is declared as follows:

  type (*fptr)( );

  This tells the compiler that fptr is a pointer to a function which returns type value

Parameter Passing Mechanisms

- Arguments are generally passed to functions in one of the two ways
  1. Call by value
     - Sending values of the arguments
     - Value is copied from argument list to parameters
     - Changes in function do not effect original variables
  2. Call by reference
     - Sending the address of the arguments
     - Passes original argument's address
     - Changes in function effect original variable

Call by Value

```
int main(int argc, char** argv) {
    int x=10;
    printf("Value of x is %d",x);
    displays(x);
    printf("\nNew Value of x is
   %d",x);
    return (EXIT_SUCCESS);
}
void displays(int  y){
   y=20;
}
```

Call by Reference

```
int main(int argc, char** argv) {
    int x=10;
    printf("Value of x is %d",x);
    displays(&x);
    printf("\nNew Value of x is
   %d",x);
    return (EXIT_SUCCESS);
}
void displays(int *y){
   *y=20;
}
```

Summary

- A pointer variable can be used to access the elements of an array of the same type
- Arrays may contain pointers
- A string is accessed via a pointer to the first character in the string
- Arguments are generally passed to functions in one of the two ways – Call by value and call by reference

Further Reading

Kernighan, B. W. and Richie, D. (1992) *The C Programming Language.* 2nd ed., New Delhi:PHI.