CIS 236 – Programming in C

Program:	**Week 10**
Points:	**20**
Chapters:	**7**

Description

Write a program that prints the values in an array and the addresses of the array's elements using four different techniques, as follows:

1. Array index notation using array name
2. Pointer/offset notation using array name
3. Array index notation using a pointer
4. Pointer/offset notation using a pointer

Example fig07_20.c will be useful as a guide for this assignment.

Learning Objectives

In this assignment, you will:

- Use functions with array and pointer arguments
- Use indexing and offset notations to access arrays

Requirements

Your code **must** use these eight functions, using these names (in addition to main):

1. printValuesNameIndex
2. printValuesPointerIndex
3. printValuesNameOffset
4. printValuesPointerOffset
5. printAddressesNameIndex
6. printAddressesPointerIndex
7. printAddressesNameOffset
8. printAddressesPointerOffset

Requirements for the printValuesNameIndex Function

Purpose:	This function prints the values in the array.
Parameter:	integer array (square bracket notation)
Algorithm:	Print the name of the function, then print the values of the input array using a loop with array index notation. You must use the parameter. Do not declare

any other variables except a loop control variable.

Return value: None

Requirements for the printValuesPointerIndex Function

Purpose: This function prints the values in the array.

Parameter: Pointer to an int array

Algorithm: Print the name of the function, then print the values of the input array using a loop with pointer index notation. You must use the parameter. Do not declare any other variables except a loop control variable.

Return value: None

Requirements for the printValuesNameOffset Function

Purpose: This function prints the values in the array.

Parameter: integer array (square bracket notation)

Algorithm: Print the name of the function, then print the values of the input array using a loop with pointer offset notation using the array name. You must use the parameter. Do not declare any other variables except a loop control variable.

Return value: None

Requirements for the printValuesPointerOffset Function

Purpose: This function prints the values in the array.

Parameter: Pointer to an int array

Algorithm: Print the name of the function, then print the values of the input array using a loop with pointer offset notation using the pointer. You must use the parameter. Do not declare any other variables except a loop control variable.

Return value: None

Requirements for the printAddressesNameIndex Function

Purpose: This function prints the address of each element of the array.

Parameter: integer array (square bracket notation)

Algorithm: Print the name of the function, then print the address of each element of the array using a loop with array index notation. You must use the parameter. Do not declare any other variables except a loop control variable. Use the appropriate format control string for printing addresses.

Return value: None

Requirements for the printAddressesPointerIndex Function

Purpose: This function prints the address of each element of the array.

Parameter: Pointer to an int array

Algorithm: Print the name of the function, then print the address of each element of the array using a loop with pointer index notation. You must use the parameter. Do not declare any other variables except a loop control variable. Use the appropriate format control string for printing addresses.

Return value: None

Requirements for the printAddressesNameOffset Function

Purpose: This function prints the address of each element of the array.

Parameter: integer array (square bracket notation)

Algorithm: Print the name of the function, then print the address of each element of the array using a loop with pointer offset notation using the array name. You must use the parameter. Do not declare any other variables except a loop control variable. Use the appropriate format control string for printing addresses.

Return value: None

Requirements for the printAddressesPointerOffset Function

Purpose: This function prints the address of each element of the array.

Parameter: Pointer to an int array

Algorithm: Print the name of the function, then print the address of each element of the array using a loop with pointer offset notation using the pointer. You must use the parameter. Do not declare any other variables except a loop control variable. Use the appropriate format control string for printing addresses.

Return value: None

Requirements for main:

1. Declare and initialize an integer array of size 5, using the values 10, 20, 30, 40 & 50.
2. Call each function with the appropriate argument.

Other Requirements for this Program

No other variables should be declared in main, above main, in any function, or in any function parameter list except those explicitly noted in the instructions.

You must a symbolic constant for the array size wherever it is needed.

All output must display exactly as it appears in the sample run. Note that your actual addresses may differ.

Sample Run

```
Printing array values from function printValuesNameIndex
a[0] = 10
a[1] = 20
a[2] = 30
a[3] = 40
a[4] = 50

Printing array values from function printValuesPointerIndex
a[0] = 10
a[1] = 20
a[2] = 30
a[3] = 40
a[4] = 50

Printing array values from function printValuesNameOffset
a[0] = 10
a[1] = 20
a[2] = 30
a[3] = 40
a[4] = 50

Printing array values from function printValuesPointerOffset
a[0] = 10
a[1] = 20
a[2] = 30
a[3] = 40
a[4] = 50

Printing array addresses from function printAddressesNameIndex
&a[0] = 62fe00
```

```
&a[1] = 62fe04
&a[2] = 62fe08
&a[3] = 62fe0c
&a[4] = 62fe10

Printing array addresses from function printAddressesPointerIndex
&a[0] = 62fe00
&a[1] = 62fe04
&a[2] = 62fe08
&a[3] = 62fe0c
&a[4] = 62fe10

Printing array addresses from function printAddressesNameOffset
&a[0] = 62fe00
&a[1] = 62fe04
&a[2] = 62fe08
&a[3] = 62fe0c
&a[4] = 62fe10

Printing array addresses from function printAddressesPointerOffset
&a[0] = 62fe00
&a[1] = 62fe04
&a[2] = 62fe08
&a[3] = 62fe0c
&a[4] = 62fe10
```

Requirements for Full Credit on This Project

SUBMIT YOUR OWN WORK – Plagiarism is not tolerated in this course. Please review the section on the Academic Honor Code in the syllabus. I will not hesitate to drop you from this class if you submit a program that is plagiarized.

COMPLETE AND ACCURATE – Your program must compile, execute, and give accurate output.

FOLLOW ALL REQUIREMENTS ACCORDING TO THE INSTRUCTIONS – Follow the instructions as written for completing this project, even if you [think you] know a "better" way to do something.

COMMENTS – Include comments in your code. There must be a comment at the top of your program that includes your name, the program number, and a description of the program. There must be comments at each important step in your program that describes that step. Every variable must include a comment describing its purpose.

BEST PRACTICES – Follow best practices in C programming as discussed in class and in the textbook, including, but not limited to, appropriate use of white space, indenting, alignment, meaningful identifier names, etc. Points will be deducted for sloppy code that is hard to read, even if it works, so pay attention to these details.

SUBMIT ONLY .c SOURCE CODE – Pay attention to the file extension of the source code file you submit. I will deduct points for not following this requirement.

SUBMIT ALL FILES BEFORE THE DUE DATE – Submit your .c source code file to the dropbox for this assignment on Canvas before the due date. Do not submit executable files. Do not submit project files from an IDE. I will not accept links to online storage.