Helwan University

Programming Language I
(Spring 2016)

Sheet 4: Pointers, Call by reference, Dynamic array

1. For each of the following, write a single statement that performs the specified task. Assume that floating-point variables number1 and number2 have been declared and that number1 has been initialized to 7.3. Assume that variable ptr is of type char *. Assume that arrays s1 and s2 are each 100-element char arrays that are initialized with string literals.
   a) Declare the variable fPtr to be a pointer to an object of type double.
   b) Assign the address of variable number1 to pointer variable fPtr.
   c) Print the value of the object pointed to by fPtr.
   d) Assign the value of the object pointed to by fPtr to variable number2.
   e) Print the value of number2.
   f) Print the address of number1.
   g) Print the address stored in fPtr. Is the value printed the same as the address of number1?
2. For each of the following, write C++ statements that perform the specified task. Assume that double-precision, floating-point numbers are stored in eight bytes and that the starting address of the array is at location 1002500 in memory. Each part of the exercise should use the results of previous parts where appropriate.
   a) Declare an array of type double called numbers with 10 elements, and initialize the elements to the values 0.0, 1.1, 2.2, …, 9.9. Assume that the symbolic constant SIZE has been defined as 10.
   b) Declare a pointer nPtr that points to a variable of type double.
   c) Use a for statement to print the elements of array numbers using array subscript notation. Print each number with one position of precision to the right of the decimal point.
   d) Write two separate statements that each assign the starting address of array numbers to the pointer variable nPtr.
   e) Use a for statement to print the elements of array numbers using pointer/offset notation with pointer nPtr.
   f) Use a for statement to print the elements of array numbers using pointer/offset notation with the array name as the pointer.
   g) Use a for statement to print the elements of array numbers using pointer/subscript notation with pointer nPtr.
   h) Refer to the fourth element of array numbers using array subscript notation, pointer/offset notation with the array name as the pointer, pointer subscript notation with nPtr and pointer/offset notation with nPtr.
   i) Assuming that nPtr points to the beginning of array numbers, what address is referenced by nPtr + 8? What value is stored at that location?
   j) Assuming that nPtr points to numbers[5], what address is referenced by nPtr after nPtr -= 4 is executed? What's the value stored at that location?

Sheet 4: Pointers, Call by reference, Dynamic array

3. Find the error in each of the following program segments. Assume the following declarations and statements:

```
int *zPtr; // zPtr will reference array z
void *sPtr = 0;
int number;
int z[ 5 ] = { 1, 2, 3, 4, 5 };
```

a) `++zPtr;`

b)
```
// use pointer to get first value of array
number = zPtr;
```

c)
```
// assign array element 2 (the value 3) to number
number = *zPtr[ 2 ];
```

d)
```
// print entire array z
for ( int i = 0; i <= 5; ++i )
cout << zPtr[ i ] << endl;
```

e)
```
// assign the value pointed to by sPtr to number
number = *sPtr;
```

f) `++z;`

4. **Quicksort,** recursive sorting technique. The basic algorithm for a single-subscripted array of values is as follows:

   I) *Partitioning Step:* Take the first element of the unsorted array and determine its final location in the sorted array (i.e., all values to the left of the element in the array are less than the element, and all values to the right of the element in the array are greater than the element). We now have one element in its proper location and two unsorted subarrays.

   II) *Recursive Step:* Perform *Step 1* on each unsorted subarray.

Each time *Step 1* is performed on a subarray, another element is placed in its final location of the sorted array, and two unsorted subarrays are created. When a subarray consists of one element, that subarray must be sorted; therefore, that element is in its final location.

5. Write a function selectionSort that implement selection sort algorithm. The first iteration of the algorithm selects the smallest element in the array and swaps it with the first element. The second iteration selects the second-smallest element (which is the smallest element of the remaining elements) and swaps it with the second element. The algorithm continues until the last iteration selects the second-largest element and swaps it with the second-to-last index, leaving the largest element in the last index. After the ith iteration, the smallest i items of the array will be sorted into increasing order in the first i elements of the array. Write a program using this function that, if the user enters 1, a pointer to function ascending is passed to function selectionSort, causing the array to be sorted into increasing order. If the user enters 2, a pointer to function descending is passed to function selectionSort, causing the array to be sorted into decreasing order.

Sheet 4: Pointers, Call by reference, Dynamic array

6. A string comparing function that should take two null terminated array of characters (s1, s2). The first character in both s1and s2 are compared. If the first character in s1 and s2 are identical the comparison between next character in s1 and s2 takes place and so on ...
   Then use this function to write a program that:
   - Asks the user how many names they wish to enter.
   - Asks the user to enter each name.
   - Calls a function to sort the names (modify the selection sort code)
   - Prints the sorted list of names.

7. Write a function getWords that takes a pointer to a null terminated array of characters (s1) and will return the list of words contained in that string.

   Hint:
   1-this list of words should be dynamically allocated array of strings.
   2-string is a null terminated array of characters.

8. Using dynamically allocated arrays, Write a function that will generate a string from its compressed version. Ex."4a2s1d" Should be transformed into "aaaassd" after allocating space for 8 characters (7 characters and one null character)