

In class Practice 10/11/2019

Group three or four people team.

Part I

1. A pointer should be initialized to nullptr or memory.
2. A pointer that is declared to be of type void* can be dereferenced. **false. no data type, so no memory.**
3. Declare a built-in 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 constant size has been defined as 10. **double numbers[10]{0.0,1,1,2.2,3.3,4.4,5.5,6.6,7.7,8.8,9.9};**
4. Declare a pointer myPtr that points to a variable of type float. **float* myPtr{1.5};**
5. Write two separate statements that each assign the starting address of built-in array numbers to the pointer variable myPtr. **myPtr = numbers; myPtr = &numbers[0];**
6. Use a *for* statement to display the elements of built-in array numbers using pointer/offset notation with the built-in array's name as the pointer.
7. Refer to the third element of built-in array call it *myGrades* using array subscript notation, pointer/offset notation with the built-in array's name as the pointer, pointer subscript notation with *myPtr* and pointer/offset notation with *myPtr*.
myGrades[2] *(myGrade + 2) myPtr[2] *(myPtr+2)
8. Assuming that *myPtr* points to the beginning of built-in array *myArr* of type *double* with starting address of the array is at location 100500 in memory, what address is referenced by *myPtr* + 8?
9. Write the function header for a function called *exchange* that takes two pointers to double-precision floating-point numbers x and y as parameters and does not return a value.

6.

```
for(size_t i = 0; i < size; i++){  
    cout << *(number + i) << ' ';  
}
```

```
8.size_t size = sizeof myPtr / sizeof(myPtr[0]);  
int address = 100500 + size*8;
```

```
9. void exchange(double* x , double* y );
```

Part II

Find the error in each of the following program segments.

Assume the following declarations and statements:

```
int number;  
int z[5]{1, 2, 3, 4, 5};  
int* zPtr; // zPtr will reference built-in array z
```

a) ++zPtr; **zPtr is nullptr now, need to initialize it.
zPtr = z; or zPtr = &z[0];**

b) // use pointer to get first value of a built-in array

```
number = zPtr;      number = *zPtr;
```

c) // assign built-in array element 2 (the value 3) to number

```
number = *zPtr[2];      number = *(zPtr+2); or number = zPtr[2]
```

d) // display entire built-in array z

```
for (size_t i{0}; i < 5; ++i)  
{ cout << zPtr[i] << endl; }
```

e) ++z; **++zPtr**

Part III

Q1. What is the mystery1 function do?

```
#include <iostream>
using namespace std;

void mystery1(char*, const char*); // prototype

int main() {
    char string1[80];
    char string2[80];

    cout << "Enter two strings: ";
    cin >> string1 >> string2;
    mystery1(string1, string2);
    cout << string1 << endl;
}

// What does this function do?
void mystery1(char* s1, const char* s2) {
    while (*s1 != '\0') {
        ++s1;
    }

    for (; (*s1 = *s2); ++s1, ++s2) {
        ; // empty statement
    }
}
```

add string2 at the end of string1

Q2 Write a C function, using pointers as parameters, which will copy a source string to the destination string. The following is the function prototype and a test driver program which use and test this function.

```
#include <iostream>
using namespace std;
```

```
char *strcpy(char *dest, const char *src);
int main()
{
    char greeting [30] = "hello, everybody";
    char echo[20] = "\0";
    cout << "Source array greeting is:" << greeting << endl;
    cout << "Before copy, echo array is:" << echo << endl;
    strcpy (echo, greeting);
    cout << "After copy greeting to echo: echo is:" << echo << endl;
    return 0;
}
```

The program generate the following results

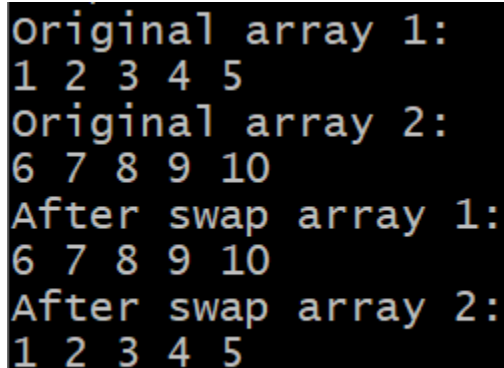
```
$ ./a
Source array greeting is:hello, everybody
Before copy, echo array is:
After copy greeting to echo: echo is:hello, everybody
```

Q3. Implement the swap function which will exchange contents of two integer arrays using pointers. The following is the function prototype and the test driver and the results of the test run.

```
#include <iostream>
using namespace std;

void swap (int* a1, int* a2, int arrSize); // function prototype
int main()
{
    int arr1 [5] ={1,2,3,4,5};
    int arr2 [5] ={6,7,8,9,10};
    size_t arrSize {5};

    cout << "Original array 1:"<<endl;
    for (int i{0}; i < arrSize; ++i)
        cout << arr1[i] <<" ";
    cout << endl;
    cout << "Original array 2:"<<endl;
    for (int i{0}; i < arrSize; ++i)
        cout << arr2[i] <<" ";
    cout << endl;
    swap (arr1, arr2, arrSize);
    cout << "After swap array 1:"<<endl;
    for (int i{0}; i < arrSize; ++i)
        cout << arr1[i] <<" ";
    cout <<endl;
    cout << "After swap array 2:"<<endl;
    for (int i{0}; i < arrSize; ++i)
        cout << arr2[i] <<" ";
    cout <<endl;
}
```



```
Original array 1:
1 2 3 4 5
Original array 2:
6 7 8 9 10
After swap array 1:
6 7 8 9 10
After swap array 2:
1 2 3 4 5
```