CPT 111 – Principles of Programming
Week 13 Programming Lab
POINTERS

Learning Outcomes:

- Describe pointer variables.
- Explain pointers in arrays.
- Demonstrate pointer arithmetic.
- Demonstrate pointers as function parameters.
- Use Dynamic Memory Allocation

---

1. Describe in C++ statement that displays the address of the variable `count`.

2. Describe in C++ the definition statement for a variable `fltPtr`. The variable should be a pointer to a `float`.

3. Given the following code.

```
int x = 7;
int *iptr = &x;
```

   Show what will be displayed if you send the expression `*iptr` to `cout`.

   Explain what happens if you send the expression `ptr` to `cout`.

4. Explain how indirection operator * works using pointer variable `ptr` and integer variable `x`.

5. Show the output of the following code.

```
int x = 50, y = 60, z = 70;
int *ptr = nullptr;
cout << x << " " << y << " " << z << endl;
ptr = &x;
*ptr *= 10;
ptr = &y;
*ptr *= 5;
ptr = &z;
*ptr *= 2;
cout << x << " " << y << " " << z << endl;
```

6. Modify the following loop so it uses pointer notation (with the indirection operator) instead of subscript notation.

```
for (int x = 0; x < 100; x++)
   cout << arr[x] << endl;
```

7. Assume `ptr` is a pointer to an `int` and holds the address 0x6ffe14. On a system with 4-byte integers, state the address that will be in `ptr` after the following statement.

```
ptr += 10;
```

8. Assume `pint` is a pointer variable. State whether each of the following statements valid or invalid. If any is invalid, explain the reason.

   A) `pint++;`

   B) `--pint;`

   C) `pint /= 2;`

   D) `pint *= 4;`

   E) `pint += x; // Assume x is an int.`

9. State whether each of the following definitions valid or invalid. If any is invalid, explain the reason.

   A)
   ```
   int ivar;
   int *iptr = &ivar;
   ```

   B) `int ivar, *iptr = &ivar;`

   C)
   ```
   float fvar;
   int *iptr = &fvar;
   ```

   D) `int nums[50], *iptr = nums;`

   E)
   ```
   int *iptr = &ivar;
   int ivar;
   ```

10. Given the following array definition.

```
int numbers[] = { 2, 4, 6, 8, 10 };
```

Show what will the following statement display.

```
cout << *(numbers + 3) << endl;
```

11. The following function uses reference variables as parameters. Modify the function so that it uses pointers instead of reference variable, then demonstrate the function in a complete program.

```
// The doSomething function
int doSomething(int &x, int &y)
{
    int temp = x;
    x = y * 10;
    y = temp * 10;
    return x + y;
}
```

12. Demonstrate a function that dynamically allocates an array of integers. The function should accept an integer argument indicating the number of elements to allocate. The function should return a pointer to the array.

13. Describe a C++ program that dynamically allocates an array large enough to hold a user-defined number of test scores. Once all the scores are entered, the array should be passed to a function that sorts them in ascending order. Another function should be called that calculates the average score. The program should display the sorted list of scores and averages with appropriate headings. Use pointer notation rather than array notation whenever possible.

    Input validation: Do not accept negative numbers for test scores.

14. Modify Question 13 above so the lowest test score is dropped. This score should not be included in the calculation of the average.

oooOOooo