

Computer Programming: Skills and Concepts

Tutorial 6(week 8): November 9 –November 14, 2009

To accompany this tutorial sheet there is an electronic file, tutw8.tar, with 3 c-programs inside. Please download (and unpack, by typing `tar -xvf tutw8.tar`) this file from the course webpage:

<http://www.inf.ed.ac.uk/teaching/courses/cp1/>

Pointer revision

Consider the following functions with the following main. Describe the relationships between integer-values and pointer-values from the various `printf` statements (including those called from within the functions).

What is the difference between variables `n` and `m`?

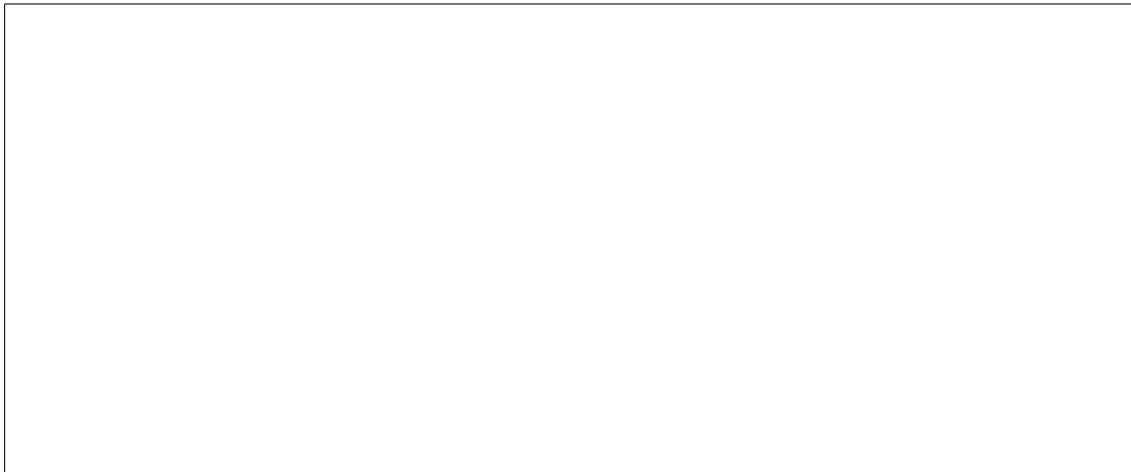
How does it manifest itself during the running of the code?

```
void addressOnly(int n) {
    printf("addressOnly: address of parameter-var n is %p, val %d.\n", &n, n);
}

void valueOfPoint(int *p) {
    printf("valueOfPoint: value of pointer p is %p, points to val %d.\n", p, *p);
}

int m=8;

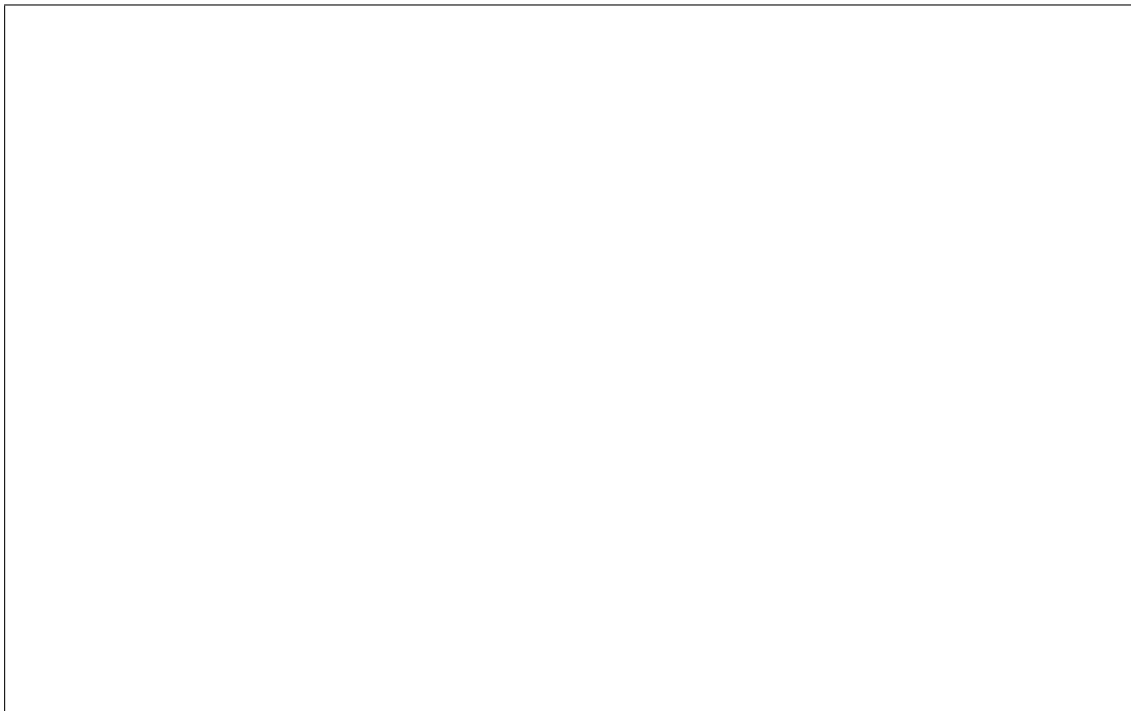
int main(void) {
    int n=5;
    printf("main: val of n is %d, address of n is %p.\n", n, &n);
    addressOnly(n);
    printf("main: Passing &n into valueOfPoint.\n");
    valueOfPoint(&n);
    printf("\n");
    printf("main: val of m is %d, address of m is %p.\n", m, &m);
    addressOnly(m);
    printf("main: Passing &m into valueOfPoint.\n");
    valueOfPoint(&m);
    return EXIT_SUCCESS;
}
```



pre-Increment vs post-Increment

In class on Thursday, 5th November, we discussed the difference between using `x++` and `++x` within another expression. Please review those slides. Now consider the `merge` function below, and write a version `merge2` which uses `++i`, `++j`, `++k` within `a[++i]`, `b[++j]`, `c[++k]` (*instead* of using `i++`, `j++`, `k++`). You will need to make some changes to make sure that `merge2` performs the merging correctly.

```
void merge(int a[], int b[], int c[], int m, int n) {
    int i=0, j=0, k=0;
    while (i < m && j < n) {
        if (a[i] <= b[j])
            c[k++] = a[i++];
        else
            c[k++] = b[j++];
    }
    /* At this stage, done comparing. All the rest of c will come
     * from either a, or else b, depending on which has stuff left. */
    while (i < m)
        c[k++] = a[i++];
    while (j < n)
        c[k++] = b[j++];
}
```



Pointers Questions

We had a discussion on pointers, functions, and arrays during a revision lecture of Tuesday 3rd Nov. Our reference point was the program `pointers.c` (linked to from the main webpage), plus an extra function `evenDivComments2` which we developed on the board (and is included in the current version of `pointers.c`).

I expect there remain many unanswered questions. Please continue the discussion in tutorial.