CSC180 tutorial notes – week of Oct. 20

For this tutorial, we mostly discuss strings.

Strings

Strings are arrays of chars with 0 at the end.

```
char str[] = “hello”;
```

<table>
<thead>
<tr>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>h</td>
<td>E</td>
<td>l</td>
<td>l</td>
<td>o</td>
<td>\n</td>
<td>‘\0’</td>
</tr>
</tbody>
</table>

- EOF (end of file is a special character (like ‘\n’)) that we can get from the input.

Some string functions:

```c
int strlen(char *s)
{
    char *p = s;
    while(*p != ‘\0’)
        p++;
    return p-s;
}
```

```c
char* strcpy(char *dest, char *src)
{
    int i;
    i = 0;
    do{
        dest[i] = src[i];
        i++;
    }while(src[i-1] != ‘\0’);
    return dest;
}
```

Note that if we wanted to use a while loop instead of do-while, we’d have to do something like:

```c
dest[0] = src[0];
i=1;
while(src[i-1] != ‘\0’){
    dest[i] = src[i];
    i++;
}
/* return <0 if s < t, 0 if s==t, >0 if s > t, in lexicographical (i.e., dictionary) order. */
int strcmp(char *s, char *t)
{
    for( ; *s == *t; s++, t++)
        if (*s == '\0')
            return 0;
    return (*s - *t);
}

int strncmp(const char *s1, const char *s2, size_t n)
{
    unsigned char uc1, uc2;
    /* Nothing to compare?  Return zero. */
    if (n == 0)
        return 0;
    /* Loop, comparing bytes. */
    while (n-- > 0 && *s1 == *s2) {
        /* If we've run out of bytes or hit a null, return zero
         * since we already know *s1 == *s2. */
        if (n == 0 || *s1 == '\0')
            return 0;
        s1++;
        s2++;
    }
    uc1 = (*(unsigned char *) s1);
    uc2 = (*(unsigned char *) s2);
    return ((uc1 < uc2) ? -1 : (uc1 > uc2));
}

cchar *strcat(char *restrict s1, const char *restrict s2)
{
    char *s = s1;
    /* Move s so that it points to the end of s1. */
    while (*s != '\0')
        s++;
    /* Copy the contents of s2 into the space at the end of s1. */
    strcpy(s, s2);
    return s1;
}

A recursive strcpy:

cchar* strcpy(char *dest, char *src)
{
    *dest = *src;
    if(*src != '\0')
        strcpy(dest+1, src+1);
    return dest;
}
Multi-dimensional arrays, arrays of pointers

Write a function that returns the number of the day of the year (e.g., Jan. 2 is always the second day of the year, Feb 1 the 32\textsuperscript{nd}, March 1 is 60\textsuperscript{th} in a non-leap year, 61\textsuperscript{st} in a leap year) given the date. (you see those on TTC transfers, for example)

An array of arrays:

```c
int day_tab[2][13] = {
    {0, 31, 28, 31, 30, 31, 30, 31, 30, 31, 30, 31, 30, 31},
    {0, 31, 29, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31, 30, 31};
```

```c
int day_of_year(int year, int month, int day)
{
    int i, leap;

    leap = year%4 == 0 && year%100 != 0 || year%400 == 0;
    for(i = 1; i < month; i++)
        day += day_tab[leap][i];

    return day;
}
```

An array of pointers:

```c
char *weeknames[] = {
    "Sunday", "Monday", "Tuesday", "Wednesday", "Thursday",
    "Friday", "Saturday"
};
```

Behaves in almost exactly the same way.
- weeknames[0] is “Sunday”
- weeknames[1][3] is ‘d’