CSC 270 – Survey of Programming Languages

C Lecture #4 Addendum - Substrings

**strcpy()**

- `strcpy(s, t)` copies the contents of `t` into `s`.
- This is standard way of assigning a value to a string, i.e., we copy it character by character instead of copying over the address at which it starts.
**strcpy() – A Example**

```c
#include <stdio.h>

int main(void)
{
    char s[80];

    strcpy(s, "Let's learn strings");
    printf("%s", s);

    return(0);
}
```

Output

Let's learn strings

---

**strncpy()**

- `strncpy(s, t, n)` copies the contents of `t` into `s` but no more than n-1 characters.
- This is standard way of assigning a value to a string, i.e., we copy it character by character instead of copying over the address at which it starts.


**strncpy() – An Example**

```c
#include <stdio.h>
#include <string.h>

int main(void)
{
    char s[80], t[80];

    strcpy(s, "The quick brown fox jumped ":
    "over the lazy dogs");
    strncpy(t, s, 10);
    /*
     * You need this in case s was more
     * than 10 characters
     */
    t[10] = '\0';

    printf("s = "s, t[80];
    printf("t = "s, t);

    return(0);
}
```
**strcat()**

- `strcat(s, t)` concatenates `s` and `t`, saving the new string in `s`.
- Example
  ```
  strcpy(s, "Robert");
  strcpy(t, "Michael");
  strcat(s, t);
  printf("s = \"%s\"\n", s);
  ```
  Output
  `s = "RobertMichael"`

**strncat()**

- `strncat(s, t, n)` concatenates `s` and `t`, saving the new string in `s`. At most it will copy `n` characters.
- Example
  ```
  strcpy(s, "Robert");
  strcpy(t, "Michael");
  strncat(s, t, 5);
  printf("s = \"%s\"\n", s);
  ```
  Output
  `s = "RobertMicha"`
```c
char text[] = "Fourscore and seven years ago";

char dest[20];
```
char *s, *t;

strncpy(dest, "abcd", 10);
memset(dest, '\0', sizeof(dest));

t = strstr(text, "score");
```c
strncpy(dest, t, 10);

t

\[\text{score and seven}\]

dest

\[\text{score and}\]  \[0\]  \[0\]  \[0\]  \[0\]  \[0\]

\[0\]  \[0\]  \[0\]  \[0\]  \[0\]

dest[10] = '\0';

t

\[\text{score and seven}\]

dest

\[\text{score and}\]  \[0\]  \[0\]  \[0\]  \[0\]  \[0\]

\[0\]  \[0\]  \[0\]  \[0\]  \[0\]

\[\text{dest[10]}\]
```
#include <string.h>
#include <stdio.h>

char text[] = "Fourscore and seven years ago";

int main(void) {
    char dest[20];
    char *s, *t;
    int i;

    strncpy(dest, "abcd", 10);
    printf("%s\n", dest);
    memset(dest, '\0', sizeof(dest));
    t = strstr(text, "score");
    strncpy(dest, t, 10);
    dest[10] = '\0';

    for (i = 0; i < 20; i++)
        putchar(dest[i]);
    putchar('\n');
    for (i = 0; i < 20; i++)
        printf("%d ", (int)dest[i]);
    putchar('\n');

    for (i = 0; i < 20; i++)
        printf("%d ", (int)t[i]);
    putchar('\n');
The output from `substring.c`

```c
printf("%s\n", t);
printf("%s\n", dest);
return(0);
```