MC140: lecture #16

today's topic:

pointers

string library

pointers.

- variables that contain memory addresses as their values
- other data types we've learned about use *direct* addressing
- pointers facilitate *indirect* addressing

declaring pointers.

- pointers indirectly address memory where data of the types we've already discussed is stored (e.g., int, char, float, double)
- declaration uses asterisk (*) to indicate a pointer to a memory location storing a particular data type
- examples:

```
int *a;
float *avg;
```

address operator.

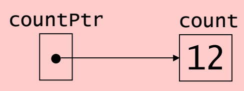
- ampersand (&) is the address operator
- it says: return the address of the variable argument
- example:

```
int count;
int *countPtr = &count;
```

&count returns the address of count and stores it in the pointer variable countPtr

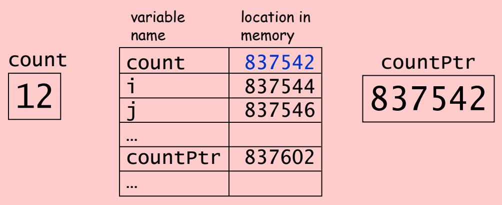
a picture.

```
int count = 12;
int *countptr = &count;
```



another picture.

```
int count = 12;
int *countptr = &count;
```



variable name	location in memory
count	837542
i	837544
j	837546
…	
countPtr	837602
…	

pointer arithmetic.

- pointers are really integers
- so you can perform integer arithmetic on them
- e.g., +1 increments a pointer, -1 decrements
- example:

```
int count = 12;
int abcde = 10;
int *countptr = &count;
count++;
countPtr++;
```

variable name	memory location	value before	value after
count	837542	12	13
abcde	837543	10	10
…			
countPtr	837602	837542	837543
…			

string library.

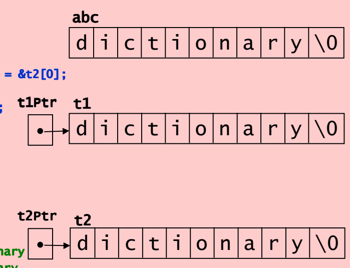
`char *` *is a pointer to a character array*

- string handling library
  `#include <string.h>`
- functions include:
  - string length
    `int strlen( char *s );`
  - string copy
  - string concatenate
  - string compare
  - string search

string library: string copy.

- **`char *strcpy( char *s1, const char *s2 );`**
- **`char *strncpy( char *s1, const char *s, size_t n );`**
- example:

```
#include <stdio.h>
#include <string.h>
int main( void ) {
  char *abc = "dictionary";
  char t1[100], t2[100];
  char *t1ptr = &t1[0], *t2ptr = &t2[0];
  t1ptr = strcpy(t1ptr,abc);
  t2ptr = strncpy(t2ptr,abc,7);
  t2[7] = '\0';
  printf( "abc=%s\n",abc);
  printf( "t1 =%s\n",t1);
  printf( "t2 =%s\n",t2);
  return( 0 );
} /* end of main() */
```



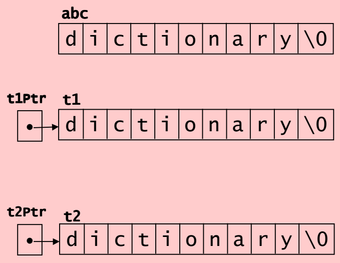
- output:

```
abc = dictionary
t1 = dictionary
t2 = diction
```

string library: string compare.

- **`char *strcmp( char *s1, const char *s2 );`**
- **`char *strncmp( char *s1, const char *s2, size_t n );`**
- example:

```
#include <stdio.h>
#include <string.h>
int main( void ) {
   . . .
   int i;
   i=strcmp(t1ptr,t2ptr);
   printf( "i=%d\n",i );
   i = strncmp(t1ptr,t2ptr,7);
   printf( "i=%d\n",i );
   . . .
} /* end of main() */
```



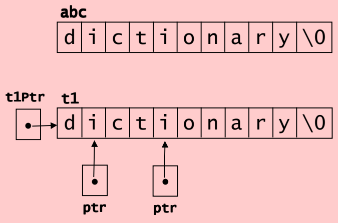
output:

```
i=0
i=1
```

string library: string search.

- **`char *strchr( char *s, int c );`**
- **`char *strrchr( char *s,int c );`**
- example:

```
#include <stdio.h>
#include <string.h>
int main( void ) {
   . . .
   char *ptr = strchr( t1,'i' );
   printf( "ptr=%s\n",ptr );
   ptr = strrchr( t1,'i' );
   printf( "ptr=%s\n",ptr );
   . . .
} /* end of main() */
```



output:

```
ptr=ictionary
ptr=ionary
```

reading.

- material covered today:
  - DD: 7.1-7.3, 8.6-8.8
- EXAM #2 will be on MON 19 MARCH
- EXAM #3 will be on WED 11 APRIL