MC140: lecture #25

today's topic:

structures

structures.

- a *structure* is a way of grouping related variables together
- syntax:

```
struct person_tag {
  char name[20];
  int  age;
} p;
```

- definition and declaration in one statement

example 1.

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

int main( void ) {

  struct person_tag {
    char name[20];
    int  age;
  } p;

  strcpy( p.name,"alex" );
  p.age = 5;

  printf( "name = %s\n",p.name );
  printf( "age  = %d\n",p.age );

  return( 0 );

} /* end of main() */
```


typedef

- lets you define your own data types
- syntax:

```
typedef int monkey;
```

- this only defines a data type; it does NOT declare a variable
- you have to do that separately
- declaration:

```
monkey m;
```


example 2.

```
#include <stdio.h>

typedef int monkey;

int main( void ) {

  monkey m;
  int    n;

  m = 3;
  n = 4;

  printf( "m = %d\n",(int)m );
  printf( "n = %d\n",n );

  return( 0 );
} /* end of main() */
```


typedef struct

- combine `typedef` and `struct` to define your own data structure
- syntax:

```
typedef struct person_tag {
  char name[20];
  int  age;
} person_type;
```

- declaration:

```
person_type p;
```

example 3.

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

typedef struct person_tag {
  char name[20];
  int  age;
} person_type;

int main( void ) {

  person_type p;

  strcpy( p.name,"alex" );
  p.age = 5;

  printf( "name = %s\n",p.name );
  printf( "age  = %d\n",p.age );

  return( 0 );
} /* end of main() */
```


pointers to structures.

- useful:
  - when passing to functions
  - in arrays
- declaration:

```
person_type *p;
```

- usage:

```
p->name
```


example 4.

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

typedef struct person_tag {
  char name[20];
  int  age;
} person_type;

void printPerson( person_type p );
void initPerson( person_type *p,
                 char *name,
                 int age );

int main( void ) {
  person_type p;
  initPerson( &p,"alex",5 );
  printPerson( p );
  return( 0 );
} /* end of main() */
```

```
void initPerson( person_type *p,
                 char *name,
                 int age ) {
  strcpy( p->name,name );
  p->age = age;
} /* end of initPerson() */

void printPerson( person_type p ) {
  printf( "name = %s\n",p.name );
  printf( "age  = %d\n",p.age );
} /* end of printPerson() */
```


arrays of structures.

- declaration:

```
person_type p[10];
```

- usage:

```
p[i].name
```


example 5.

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

typedef struct person_tag {
  char name[20];
  int  age;
} person_type;

void printPerson( person_type *p,
                  int size );
void initPerson( person_type *p,
                 char *name,
                 int age );

int main( void ) {
  person_type p[3];
  initPerson( &p[0],"alex",5 );
  initPerson( &p[1],"jen",9 );
  initPerson( &p[2],"suze",12 );
  printPerson( p,3 );
  return( 0 );
} /* end of main() */
```

```
void initPerson( person_type *p,
                 char *name,
                 int age ) {
  strcpy( p->name,name );
  p->age = age;
} /* end of initPerson() */

void printPerson( person_type p[],
                  int size ) {
  int i;
  for ( i=0; i<size; i++ ) {
    printf( "person #%d\n",i );
    printf( "name = %s\n",p[i].name );
    printf( "age  = %d\n",p[i].age );
    printf( "\n" );
  } /* end for i */
} /* end of printPerson() */
```


reading.

- DD: 10.1-10.6
- FINAL EXAM: Monday May 7, 4pm