Topics: Arrays, Pointers, Functions

Approach: Introduce Pointers, Explore their use

Main Ideas:

Quick Review - Forms, Connector, Scripts, Tools in C
- Focus tonight on memory usage in C

Recall Memory
- A sequence of boxes, each numbered
- All data and code live in memory
- Today we learn to program with the addresses of memory

Why?
- Pass by reference : useful!
- Linked data structures: Really useful
- dynamic memory : super useful
- other reasons : no so important

Types of storage:
- single values : char, int, float
- array: contiguous sequence of one type
- struct: varied types in one container

Single values:
- Where are the values stored?
  - ex1pa.c -- print the addresses
  - ex1sa.c -- store the addresses
- What can we do with pointers?
  - ex1dp.c -- dereference pointes
  - ex1cp.c -- compare pointers
  - ex1pf.c -- pass pointers to functions
- Question: Do pointers have addresses?

Arrays:
- ex2.c -- take address, deref, compare
- What can we do with pointers to arrays?
  - ex2ia.c -- index into array using [] notation
  - ex2ao.c -- arithmetic (++,--,+,-)
  - ex2ae.c -- more exercises -- trace these

Structs:
- ex3.c -- pointers and structs
- What can we do?
- Take addresses, compare, assign, select members
- pass to functions by reference

Arrays of Pointers
- ex4.c -- what does this code do?
- Draw a picture.

```
:::::::::::::: ex1pa.c ::::::::::::::
/* ex1pa.c: print addresses */
#include <stdio.h>

typedef unsigned long ul;

int main()
{
     int i,j;
     char c;
     float a;

     i = 2;
     j = i;
     if ( i == j )
        c = 't';

     printf("i=%d, j=%d, c=%c, a=%f\n", i,j,c,a);
     printf("locations are:\n");
     printf("i=%p, j=%p, c=%p, a=%p\n", &i,&j,&c,&a);
     printf("i at %lu, j at %lu, c at %lu, a at %lu\n",
               (ul)&i,(ul)&j,(ul)&c,(ul)&a);
     return 0;
}
:::::::::::::: ex1sa.c ::::::::::::::

/* ex1sa.c: store addresses */

#include <stdio.h>

typedef unsigned long ul;

int main()
{
     int i,j;
     char c;
     float a;

     int *p;    /* p holds address of an int */
     int *q;    /* q holds address of an int */
     char *cp;

     p = &i;    /* get address and store it  */
     q = &j;    /* get address and store it  */
     cp = &c;

     i = 3;
     j = i;
     if ( i == j )
        c = 't';

     /* now to print them out */

     printf("i=%d, j=%d, c=%c\n", i,j,c);
     printf("locations are:\n");
     printf("  in hexadecimal:\n");
     printf("    i at %p, j at %p, c at %p\n", p,  q,  cp);
     printf("  in decimal:\n");
     printf("    i at %lu, j at %lu, c at %lu\n", (ul)p,  (ul)q,  (ul)cp);
     printf("    p at %lu, q at %lu, cp at %lu\n", &p,  &q,  &cp);
}
```

```
:::::::::::::: ex1dp.c ::::::::::::::

/* ex1dp.c: dereference pointers */

#include <stdio.h>

typedef unsigned long ul;

int main()
{
     int i,j;
     char c;

     int *p;    /* p holds address of an int */
     int *q;    /* q holds address of an int */
     char *cp;

     p = &i;    /* get address and store it  */
     q = &j;    /* get address and store it  */
     cp = &c;

     *p = 3;   /* same as: i = 3; */
     *q = *p;  /* same as: j = i; */
     if ( *p == *q )   /* same as if ( i == j ) */
        *cp = 't' ;    /* same as       c = 't'; */

     /* now to print them out */

     printf("i=%d, j=%d, c=%c\n", i,j,c);
     printf("locations are:\n");
     printf("i at %p, j at %p, c at %p\n", p, q, cp);
}
:::::::::::::: ex1cp.c ::::::::::::::

/* ex1cp.c: compare pointers */

#include <stdio.h>

typedef unsigned long ul;

int main()
{
     int i,j;
     char c = 'x';

     int *p;    /* p holds address of an int */
     int *q;    /* q holds address of an int */
     char *cp;

     p = &i;    /* get address and store it  */
     q = &j;    /* get address and store it  */
     cp = &c;

     *p = 3;   /* same as: i = 3; */
     *q = *p;  /* same as: j = i; */
     if ( *p == *q ){  /* same as if ( i == j ) */
        printf("*p equals *q: values pointed to are equal\n");
        *cp = 't' ;    /* same as       c = 't'; */
     }
     if ( p == q )
        printf("p equals q: p and q point to same location\n");
     else
        printf("p != q: p and q point to different places\n");

     /* now to print them out */
     printf("i=%d, j=%d, c=%c\n", i,j,c);
     printf("locations are:\n");
     printf("i at %p, j at %p, c at %p\n", p, q, cp);
     return 0;
}
```

```
:::::::::::::: ex1pf.c ::::::::::::::

/* ex1pf.c: pass to functions */

#include <stdio.h>

void compare(int *, int *);
void display(int *, char *);

int main()
{
     int i,j;
     char c;

     int *p;    /* p holds address of an int */
     int *q;    /* q holds address of an int */
     char *cp;

     p = &i;    /* get address and store it  */
     q = &j;    /* get address and store it  */
     cp = &c;

     *p = 3;   /* same as: i = 3; */
     *q = *p;  /* same as: j = i; */
     if ( *p == *q )
         c = 't';

     compare(p, q);
     display(p, cp);
}
/*
 * compare values AND addrs of two int ptrs
 */
void compare(int *p1, int *p2)
{
     if ( *p1 == *p2 )        /* same as if ( i == j ) */
        printf("values of pointees are equal\n");
     if ( p1 == p2 )
        printf("both point to same place\n");
     else
        printf("point to different places\n");
}
void display(int *ip, char *cp)
{
    printf("values are: %d %c\n", *ip, *cp);
    printf("addrs  are %p %p\n",   ip,  cp);
}
:::::::::::::: ex2.c ::::::::::::::
/* ex2.c: arrays */

#include <stdio.h>

int main()
{
        char    m[10] = "hello";
        char    n[]   = "how are you?";

        char    *p, *q;
        p = &m[0];    /* OR p = m */
                      /* name of array is addr of 1st el */
        q = n;

        if ( *p == *q )
                printf("values are same\n");
        if ( p == q )
                printf("addresses are same\n");
        else
                printf("addresses differ\n");

        printf("the char at m is %c\n", *m);
        printf("the string at m is %s\n", m);
        printf("the address of m is %lu\n", (unsigned long) m);
        printf("the address of n is %lu\n", (unsigned long) n);
        return 0;
}
```

```
:::::::::::::: ex2ia.c ::::::::::::::
/* ex2ia.c index into arrays */

#include <stdio.h>

int main()
{
        char    m[10] = "hello";
        char    n[]   = "how are you?";

        char    *p, *q;
        p = &m[0];    /* OR p = m */
        q = n;

        printf("chars in m[2] and m[4] are %c %c\n", p[2], p[4]);
        *q = p[2];
        printf("string at n is %s\n", q);
}
:::::::::::::: ex2ao.c ::::::::::::::
/* ex2ao.c: arithmetic operations */

#include <stdio.h>

int main()
{
        char    m[10] = "hello";
        char    n[]   = "how are you?";

        char    *p, *q;
        p = &m[0];    /* OR p = m */
        q = n;

        printf("chars in m[2] and m[4] are %c %c\n", p[2], p[4]);
        printf("string at n is %s\n", q);

        p++;
        q = q + 3;
        printf("chars in m[2] and m[4] are %c %c\n", p[2], p[4]);
        printf("string at n is %s\n", q);
}
:::::::::::::: ex2ae.c ::::::::::::::
/* ex2ao.c: arithmetic exercises */
#include <stdio.h>
/*
    232  236 240  244   248  252  256  260     addrs
   +----+----+----+----+----+----+----+----+----+----+----+----+----+----+
   |    |    |    | ?  |  ?  |  ?  |  ?  |  ?  |    |    |    |    |    |
   +----+----+----+----+----+----+----+----+----+----+----+----+----+----+
                  +---------+
                p |         |
                  +---------+
                 +---+
                i| ? |
                 +---+
 */
int main()
{
        int     t[5], *p, i;

        p = t;
        for ( i=0 ; i<5 ; i++ ){
                *p = i ;
                p++;
        }
        for(i=0; i<5; i++){
                printf("t[%d] = %d\n", i, t[i]);
        }
        printf("t is at %p, p holds value %p and points to %d\n", t, p, *p);
        printf("(p+2)[0] = %d\n", (p+2)[0]);
        printf("(p-2)[4] = %d\n", (p-2)[4]);
        printf("(p+10)[-8] = %d\n", (p+10)[-8]);
        printf("(p+*p)[*(p+1)] = %d\n", (p+*p)[*(p+1)]);
        return 0;
}
```

```
:::::::::::::: strchr.c ::::::::::::::
#include <stdio.h>

char *my_strchr(char *s, char c);

typedef unsigned long ul;

/*
 * a version of strchr
 */
#define LEN     100

int main()
{
        char    line[LEN];
        char    chr;
        char    *where;

        printf("Enter some text: ");
        fgets(line, LEN, stdin);
        printf("Enter a char: ");
        chr = getchar();

        where = my_strchr(line, chr);
        printf("Line is at %lu, strchr returned %lu\n", (ul) line, (ul) where);
        return 0;
}

/* strchr(str, chr) returns address of first instance of chr in str or NULL */
char *my_strchr(char *str, char chr)
{
        // search for chr in str, return address of where chr is found
        // or return NULL if not found
        // version1: Indexing
        int     i;
        for ( i = 0 ; str[i] != '\0'; i++ )
                if ( str[i] == chr )
                        return &str[i];
        return NULL;
        //
        // version2: Using pointers
        //
}
:::::::::::::: ex3.c ::::::::::::::
/* ex3.c: pointers to structs */
#include <stdio.h>
#include <string.h>

struct time
{
        int     hr, mn;
};

struct tstop
{
        char    stn[20];
        char    dir;
        struct time when;
};

void print_event(struct tstop *);
```

```
int main()
{
        struct tstop s1, s2;          // get space for the data
        struct tstop *p1, *p2;        // get space to hold addresses

        p1 = &s1;                     // store addresses of data
        p2 = &s2;                     // in pointer variables

        strcpy(s1.stn, "lynn");               // use struct_name.member
        s1.dir = 'i';                 // when using name of struct
        s1.when.hr = 9;
        s1.when.mn = 23;

        strcpy(p2->stn, "salem");     // use struct_ptr->member
        p2->dir = 'i';                // when using ptr to struct
        p2->when.hr = 9;
        p2->when.mn = 12;

        print_event( &s1 );           // pass address struct s1
        print_event( p2  );           // p2 holds address of struct s2
        return 0;
}

/*
 * print out an event: arg is pointer to a train stop event
 */
void print_event(struct tstop *p)
{
        printf("station: %s\n", p->stn);
        printf("    dir: %c\n", p->dir);
        printf("   when: %d:%d\n", p->when.hr, p->when.mn);
}

:::::::::::::: ex4.c ::::::::::::::
/* ex4.c: what do I do? */
#include <stdio.h>

int main()
{
        char    veg[4][20];
        int     i;

        strcpy(food[0], "peas");
        strcpy(food[1], "carrots");
        strcpy(food[2], "kale");
        strcpy(food[3], "lettuce");

        for(i=0; i<4; i++)
        {
                printf("item %d is %s\n", i, food[i]);
        }
        what_do_i_do(veg);
}

void what_do_i_do(char a[4][20])
{
        char    *p[4];
        int     i;

        for(i=0 ; i<4 ; i++)
        {
                p[i] = &a[4-i];
        }
        for(i=0 ; i<4 ; i++ )
        {
                printf("in array p, item %d is %s\n", i, p[i] + i);
        }
}
```