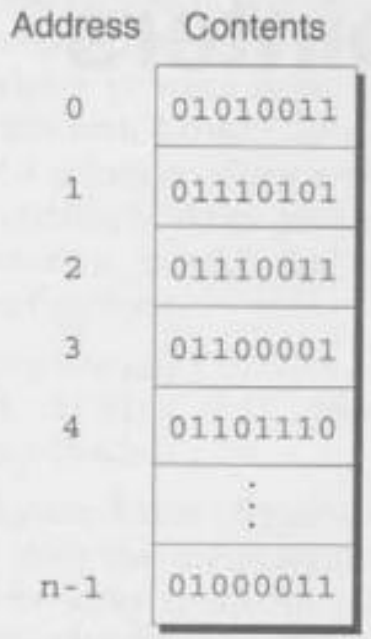
Program Organization

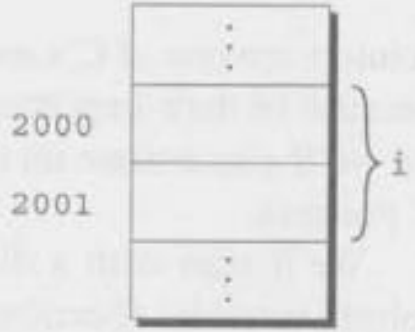
Self Study

Pointers

Pointer Variables



the address of the first byte is said to be the address of the variable. In the following figure, the variable `i` occupies the bytes at addresses 2000 and 2001, so `i`'s address is 2000:



Declaring Pointer Variables

```
int i, j, a[10], b[20], *p, *q;
```

In this example, `i` and `j` are ordinary integer variables, `a` and `b` are arrays of integers, and `p` and `q` are pointers to integer objects.

C requires that every pointer variable point only to objects of a particular type (the ***referenced type***):

```
int *p;       /* points only to integers    */
double *q;    /* points only to doubles     */
char *r;      /* points only to characters */
```

Pointer Assignment

```
int i, j, *p, *q;
```

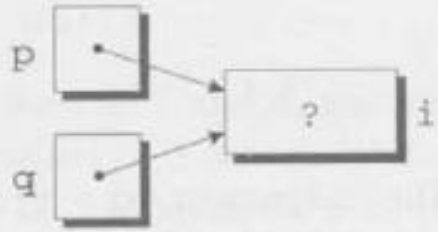
The statement

```
p = &i;
```

is an example of pointer assignment; the address of `i` is copied into `p`. Here's another example of pointer assignment:

```
q = p;
```

This statement copies the contents of `p` (the address of `i`) into `q`, in effect making `q` point to the same place as `p`:



Pointers as Arguments

```
void decompose(double x, long *int_part, double *frac_part)
{
  *int_part = (long) x;
  *frac_part = x - *int_part;
}
```

The prototype for decompose could be either

```
void decompose(double x, long *int_part, double *frac_part);
```

or

```
void decompose(double, long *, double *);
```

We'll call decompose in the following way:

```
decompose(3.14159, &i, &d);
```

PROGRAM Finding the Largest and Smallest Elements in an Array

To illustrate how pointers are passed to functions, let's look at a function named `max_min` that finds the largest and smallest elements in an array. When we call `max_min`, we'll pass it pointers to two variables; `max_min` will then store its answers in these variables. `max_min` has the following prototype:

```
void max_min(int a[], int n, int *max, int *min);
```

A call of `max_min` might have the following appearance:

```
max_min(b, N, &big, &small);
```

```
void f(const int *p)
{
  *p = 0;    /*** WRONG ***/
}
```

Constant to Protect Argument

Pointer as Return Values

The following function, when given pointers to two integers, returns a pointer to whichever integer is larger:

```
int *max(int *a, int *b)
{
  if (*a > *b)
    return a;
  else
    return b;
}
```