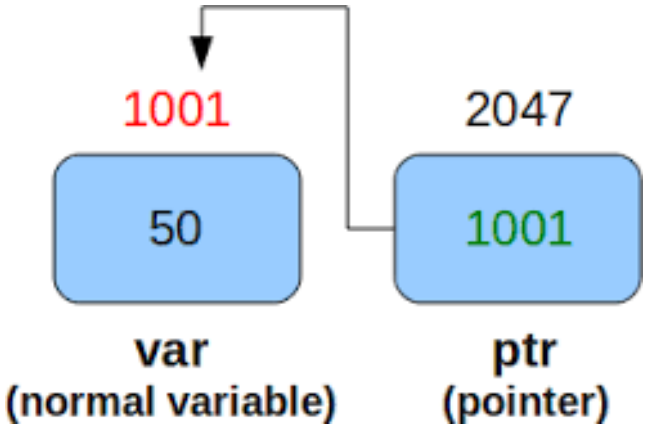
Pointers in C



A *pointer* is a variable that contains the address (location) of another variable in memory.

For expert users of C, **pointers** offer the following advantages:

1. Pointers reduce the length and complexity of a program.
2. They increase execution speed.
3. A pointer enables us to access a variable that is defined outside the function.
4. Pointers are more efficient in handling the data tables.
5. Can save space in memory.

Here we will look at some *basic* principles involving **pointers.**

Helpful things to remember with pointers:

& symbol is used to get the **address** of the variable

***** Symbol gives **value** of the variable at pointer's address

The following example includes **NO** POINTERS but demonstrates how the **&** can be used to get the ADDRESS of a variable.

```
#include <stdio.h>
int main()
{
  int var = 5;
  printf("Value: %d\n", var);
  printf("Address: %u", &var);  //Notice, the ampersand(&) before var.
  return 0;
}
```

Output

```
Value: 5
Address: 2686778
```

//the %u is the specifier for "unsigned integer" (will output address in integer form)

Using pointers

How to declare a pointer?

```
int    *p1      //Pointer to an integer variable
double *p2     //Pointer to a double variable
char   *p3     //Pointer to a character variable
float  *p4    //pointer to a float variable
```

Exercise 8.1

In the code *below* a **variable** (`int c;`) is created.

and a **pointer** is created (`unsigned int *ptr1;`). The *address* and *contents* are then displayed in **two** separate ways.

Type the code into the online compiler and **see if you can predict the output**

```
#include <stdio.h>
int main()
{
 unsigned int *ptr1;      // Pointer created
 int c;               // variable created
   c=22;
   printf("Address of c:%u\n",&c);
   printf("Value of c:%d\n\n",c);

   ptr1=&c;
   printf("Address stored in pointer:%u\n",ptr1);
   printf("Value at pointer:%d\n\n",*ptr1);

   c=11;
   printf("Address of c :%d\n",&c);
   printf("Content of c:%d\n\n",c);

   ptr1=&c;
   printf("Address stored in pointer:%d\n",ptr1);
   printf("Value at pointer:%d\n\n",*ptr1);
   return 0;
}
```

Pointers and arrays

In C programming, the **name of the array** always **points to address of the first element** of an array.

Examples:

```
Int arr[4];
```

in the above example, **`arr`** and **`&arr[0]`** both will point to the address of the first element.

`&arr[0]` is equivalent to **`arr`** **[represent the same Address]**

Also,

`arr[0]` is equivalent to **`*arr`** [**VALUES** at address the same]

Examples:

```
&arr[1] is equivalent to (arr + 1) AND, arr[1] is equivalent to *(arr + 1).
&arr[2] is equivalent to (arr + 2) AND, arr[2] is equivalent to *(arr + 2).
&arr[3] is equivalent to (arr + 3) AND, arr[3] is equivalent to *(arr + 3).
```

As you can see from above, in C, you can declare an array and can **use pointers** to collect, alter, or display the data of an array.

Exercise8.1 (type into online compiler and run)

The program below uses **pointers** to find the sum of six numbers within an array

```
#include <stdio.h>
int main()
{
  int i, classes[6],sum = 0;
  printf("Enter 6 numbers:\n");
  for(i = 0; i < 6; i++)
  {
      // (classes + i) is equivalent to &classes[i]
      scanf("%d",(classes + i));

      // *(classes + i) is equivalent to classes[i]
      sum = sum + *(classes + i);
  }
  printf("Sum = %d", sum);
  return 0;
}
```

Output

```
Enter 6 numbers:
2
3
4
5
3
4
Sum = 21
```

Key point: **`*(classes +i)`** same as **`classes[i]`**

Exercise8.2 (type into online compiler and run)

The program prints out the element of an array using **pointers**.

```
#include <stdio.h>

int main () {

   /* an array with 5 elements */
   double balance[5] = {1000.0, 2.0, 3.4, 17.0, 50.0};
   double *p;
   int i;

   p = balance;

   /* output each array element's value */
   printf( "Array values using pointer\n");

   for ( i = 0; i < 5; i++ ) {
      printf("*(p + %d) : %f\n",  i, *(p + i) );
   }

   printf( "Array values using balance as address\n");

   for ( i = 0; i < 5; i++ ) {
      printf("*(balance + %d) : %f\n",  i, *(balance + i) );
   }

   return 0;
}
```

Key point: **`*(balance +i)`** same as **`balance[i]`**

Exercise8.3 (Try then Check...solution on next page)

Write a program that:

a) asks the user for a size of an array
b) Takes in values from the user to populate the array
c) Then finds the average of the values input
d)You must use a **pointers** to both collect and calculate the average.

Use the following declarations to get you started:

```
int i;
int size;
float sum;
float avg;
int *a;     // name your pointer "a"

sum = avg = 0;

printf("Enter array size:\n");
scanf("%d",&size);

int array[size];

a=array;     // a will point to first element in array
```

Exercise 8.3 solution

```
#include <stdio.h>

int main()
{

   int i;
   int size;
   float sum;
   float avg;
   int *a;

   sum = avg = 0;

    printf("Enter array size:\n");
     scanf("%d",&size);

     int array[size];
       a=array;

     printf("Enter elements:\n");
  for(i = 0; i < size; i++)
    {
      scanf("%d",(a + i));
    }



   for(i = 0; i < size; i++)
   {
      sum = sum + *(a+i);
   }

   avg = sum/size;

   printf("Average of array values is %.2f", avg);

   return 0;
}
```

Exercise 8.4 (try then check)

Write a C program to find **the maximum value of an array from values input by the user.**
The program will:

a) Ask the user to input the number of data values in the program.
b) Ask the user to input values
c) Return the max value
d) Use ***array** to refer to your array addresses ***not*** array[i]

Sample input/putput

Enter number of data values: 3

Enter value 1: 21
Enter value 2: 12
Enter value 3: 4

The max is 21!

Use:

```
    int n;     // number of pieces of data
    int i;     // index to count through array
int arr[n];    // name your array "arr"

when referring to the values inside the array use:

*arr
*arr +1
```

Exercise 8.4 (Sample solution)

```
#include<stdio.h>

int main()
{
    int n;
    int i;

    printf("Enter number of data values");
    scanf("%d",&n);

    int arr[n];

    for(i=0;i<n;i++)
    {
      printf("Enter value %d:",i+1);
      scanf("%d",(arr+i));

    }

    for(i=1;i<n;i++)
        {
         if(*arr< *(arr+i))
         {
             *arr= *(arr+i);
         }
        }

    printf("The max value is: %d",*arr);

    return 0;
}
```