The University Of Sheffield.

Building a Well Structured Program

Using functions and managing memory

The University Of Sheffield.

Learning Outcomes

- Using and Writing Functions
- Memory management
  - Introduction to pointers
- Function calls
- Arrays and Structures
- Dynamic memory allocation
- Understanding function calls - the function pointer

The University Of Sheffield.

Functions

- **Group -** functions enable grouping of commonly used code into a reusable and compact unit.
- **Modularise -** programs containing many functions main should be implemented as a group of calls to functions undertaking the bulk of the work
- **Reuse -** become familiar with rich collections of functions in the ANSI C standard library
- **Portability -** using functions from ANSI standard library increases portability

The University Of Sheffield.

Overview the input output functions

- Printf
- Scanf
- Format specifiers
- The address operator used in scanf

The University Of Sheffield.

printf

- Provides formatted input and output
- Input for printf is a format specification followed by a list of variable names to be displayed
- Note the use of escape characters e.g. \n generates a newline

printf(“variable %d is %f\n”, myint, myfloat);

Examples

```
/*Use the printf function to Display a welcome message on the users screen*/
printf("Welcome to the C Language!\n);"
```

```
printf("b=%f fb=%f\n",b,fb);
```

The University Of Sheffield.

scanf

- Provided an input format and a list of variables
- scanf(“%d”, &myint);
- Note variable name has & in front

```
/*Request input from the user*/
printf("Enter the first integer\n");
scanf("%d", &i1);       /*Read in the integer*/
printf("Enter the second integer\n");
scanf("%d", &i2);
```

The University Of Sheffield.

Escape Characters

Escape Sequence	Description
\n	Newline, position cursor at the start of a new line
\t	Horizontal tab, move cursor to the next tab stop
\r	Carriage return. Position cursor to the beginning of the current line; do not advance to the next line.
\a	Alert, sound system warning beep
\\	Backslash, print a backslash character in a printf statement
\”	Double quote print a double quote character in a printf statement.

The University Of Sheffield.

Format Specifiers for printf and scanf

Data Type	Printf specifier	Scanf specifier
long double	%Lf	%Lf
double	%f	%lf
float	%f	%f
unsigned long int	%lu	%lu
long int	%ld	%ld
unsigned int	%u	%u
int	%d	%d
short	%hd	%hd
char	%c	%c

The University Of Sheffield.

Practice Session

Modify the root-newton example by making it compute the root of the function

x^3-2x-5

With the derivative function

$3x^2-2$

```
  while( fabs(b-a)>(FLT_EPSILON*b))
  {
    x = (a+b)/2;
    /*The function whose root is to be determined*/
    fx = pow(x,3)-2*x-5;
    if(sign(fx)==sign(fa))
    {
      a = x;
      fa = fx;
printf("a=%f fa=%f\n",a,fa);
    }
    else
    {
      b = x;
      fb = fx;
printf("b=%f fb=%f\n",b,fb);
    }

  }
  printf(" The root is :%f\n",x);
```

The University Of Sheffield.

Standard Library Functions

Header	Description
<stdio.h>	Functions for standard input and output
<float.h>	Floating point size limits
<limits.h>	Contains integral size limits of system
<stdlib.h>	Functions for converting numbers to text and text to numbers, memory allocation, random numbers, other utility functions
<math.h>	Math library functions
<string.h>	String processing functions
<stddef.h>	Common definitions of types used by C

The University Of Sheffield.

Functions in the library math.h

Function	Returns
sqrt(x)	Square root
exp(x)	Exponential function
log(x)	Natural logarithm (base e)
log10(x)	Logarithm (base 10)
fabs(x)	Absolute value
pow(x,y)	X raised to the power of y
sin(x)	Trignometric sine (x in radians)
cos(x)	Trignometric cosine (x in radians)
tan(x)	Trignometric tangent (x in radians)
atan(x)	Arctangent of x (returned value is in radians)

The University Of Sheffield.

Using Functions

- Include the header file for the required library using the preprocessor directive

  #include <libraryname.h>

- Note no semi colon after this
- Variables defined in functions are local variables
- Functions have a list of parameters

  Means of communicating information between functions

- Functions can return values
- printf and scanf good examples of function calls
- Use the –lm option to compile an application using math library functions e.g.

  gcc myprog.c –o myprog -lm

The University Of Sheffield.

Practice

- Build and run the example function1.c
- Add more calls to the blorf() function in the main pprogram
- Build and run function2.c
- Note this avoids the use of the function prototype
- Move the soup function after the main function compile and run what happens?
- Add a prototype and build and run again

The University Of Sheffield.

Practice

1. Modify the root finding examples for the newton and bisection method to call a function defined by a c- function (rather than inline as performed in the code example)
2. Compile and run functions.c.

   Run the program several times and observe that it always provides. The same output.

   Seed the random number generator using the statement srand(time(NULL));

   Run the program several times and observe the output

The University Of Sheffield.

Pointers and Arrays

Pointers are a powerful feature of C used for managing data in memory and the memory addresses of that data

- Arrays
- Strings
- Structures
- Complex data types e.g. stacks, linked lists, queues etc

The University Of Sheffield.

Review the Steps of Variable Declaration

- A variable is an area of memory that has been given a name.
- The variable declaration
  - float f1;
  - This is the command to allocate an area of memory for a float variable type with the name f1.
- The statement
  - f1=3.141
  - is a command to assign the value 3.141 to the area of memory named f1.

The University Of Sheffield.

What is a Pointer

- **Pointers are variables that contain memory addresses as their values.**
- Pointer declared using the indirection or de-referencing operator *
- Example
  - float *f1ptr;
- f1ptr is pointer variable and it is the memory location of a float variable

The University Of Sheffield.

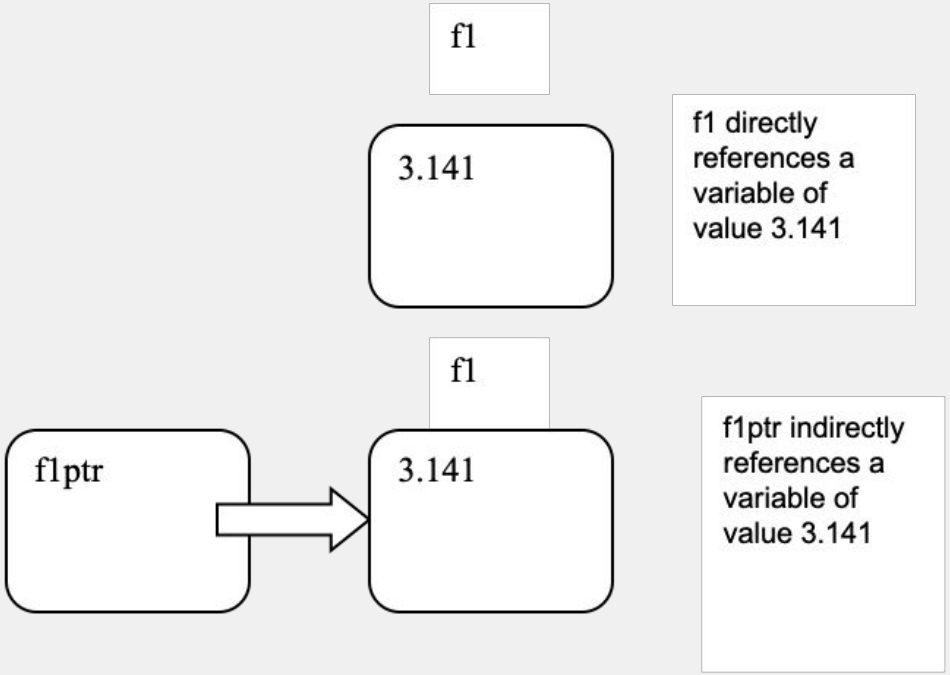
Using the Pointer Operators

- The redirection operator returns the address of a variable
- & applied to f1 returns the address of f1

```
float f1;
float *f1ptr;  /* Declare a pointer variable to an
integer*/
f1=3.141;
f1ptr=&f1; /*f1ptr is set to the address of f1*/
```

The University Of Sheffield.

Pointer Variables



The University Of Sheffield.

Using the Pointer Operators

```
int some_var;   /*1*/
int *ptr_to_some_var; /*2*/
ptr_to_some_var = &some_var; /*3*/
printf ("%d\n\n", *ptr_to_some_var); /*4*/
```

- /*1*/ Declare an integer
- /*2*/ Declare a pointer
- /*3*/ Assign a value to the pointer variable
- /*4*/ Use the pointer in a function (dereference the value)
- Compile and run the example pointers.c

The University Of Sheffield.

Function Calls

- Call by value
  - Copy of variable passed to function
  - If that variable is modified within the function then upon return from the function since only the copy has been modified, the actual variable is not modified
- Call by reference
  - Pass the address of a variable (i.e. a pointer) to a function
  - The variable pointed to can be modified within that function

The University Of Sheffield.

Call By Value

- finval=FuncByValue(finval);

```
float FuncByValue(float fval)
{
  return fval*fval;
}
```

The University Of Sheffield.

Call by Reference

- FuncByReference(&finref), Use & to pass the address of a variable to the function;
- Value of the referenced variable passed to the function is modified after returning from the function.

```
void  FuncByReference(float *fvalptr)
{
  *fvalptr = *fvalptr * *fvalptr;
}
```

The University Of Sheffield.

Initialising and Creating an Integer Array

- Initialisation
  - int iarray[5]={1,2,3,4,5};
- Or… initialise elements individually
- Note first element is referenced using 0
  - iarray[0]=1;
  - ……..
  - iarray[4]=5;

The University Of Sheffield.

Demonstration

- Putting it all together
  - Function calls
  - Simple array examples
- Numerical Method Examples
  - Numerical differentiation
  - Numerical Integration

The University Of Sheffield.

Practice

Compile and run the following programs (see the pointers folder)

a. array.c initialising and using arrays with pointers

b. bubblesort.c

Bubble sort example, using call by reference to manipulate data passed into a function c. arrayref.c Using pointer notation to manipulate arrays

The University Of Sheffield.

Practical examples

- Compile and run the following programs
  - Numerical Differentiation
    - 2 and four point methods
  - Numerical Integration
    - Trapezium method
    - Simpsons rule (includes lagrange interpolation function)

The University Of Sheffield.

Multidimensional Array

- Using the variable of a particular Type, the declaration for a multi dimensional array is made as follows:
  - *Type variable[size1][size2];*
- To access or assign an element to an element of a multidimensional array we use the statement:
  - *variable[index1][index2]=avalue;*

The University Of Sheffield.

Matrix Initialisation

- Alternatively the bracket initialisation method can be used, for example the integer matrix[2][4] can be initialised as follows:

```
int matrix[2][4]
  {
  {1,2,3,4},
  {10,20,30,40}
  };
```

The University Of Sheffield.

Arrays are pointers

- The array variable is a pointer whose value is the address of the first element of the array.
- For a one dimensional array access a value using the following pointer notation:

```
int ielement= *(iarray+1);
```

- This assignment increments the array pointer to the second element in the array (the first element is always index 0)
- uses the * operator to dereference the pointer

The University Of Sheffield.

Pointer Arrays

- A string is a pointer to an array of characters
- An array of strings is an array of pointers

Multidimensional array is essentially an array of pointer arrays

The University Of Sheffield.

Memory Leaks

- TAKE VERY SPECIAL CARE IN USE OF POINTERS AND MANAGEMENT OF ARRAYS
- A common problem when using arrays is that the program might run off the end of the array particularly when using pointer arithmetic.
- When passing an array to a function it is good practice to pass the size of that array making the function more general.

The University Of Sheffield.

Practice

- Compile and run the following programs
  - Program array.c initialising and using arrays with pointers
  - Program bubblesort.c is a bubble sort example, using call by reference to manipulate data passed into a function
  - Program arrayref.c uses pointer notation to manipulate arrays

The University Of Sheffield.

Practice

Modify the integration examples to compute the error function, defined by

$$f(x) = \frac{2}{\sqrt{\pi}} \int_0^x e^{-x^2}$$

Modify the program so that erf(x) is computed for a range of values