

C Programming Short Course

Will Jones, Nathan DeBardleben, Phil Carns

Parallel Architecture Research Laboratory
Clemson University : ECE L 371 C Short Course
Fall 1999

C program Structure

- Before the main() procedure
 - ❑ Include the correct header files – the *.h files
 - ❑ Include any pound defines global constants
 - ❑ Include function prototypes
 - ❑ Include global variables – int x; float temp;

C program Structure

- The main() Procedure
 - ❑ Define local variables and data structures
 - ❑ This is where the work is done
- After the main() procedure
 - ❑ This is where you define functions that you use in the main() procedure
- Lets take a look at an example C program format

Standard Input / Output

```
// Print to the screen
int a = 5;
printf("a is equal to the number: %i \n", a);
// Get an input from the user
int choice;
scanf("%i",&choice);

/* Here the "%i" means it will interpret what the user enters
   as an integer. Also the "&choice" indicates that the
   function scanf() needs a pointer to the variable "choice".
*/
```

Commenting Your Code

```
/* You can use comments like this to  
   select entire blocks of code or  
   for writting large comments.  
   int a = 0;  
*/
```

OR

```
// You can just comment out one line of code  
// or for one line comments.
```

Basic C Constructs

- loops – used of an iterative process
 - ❑ for loops
 - ❑ while loops
 - ❑ do while loops
- if statements – used to make decisions
 - ❑ if(condition) do something;
 - ❑ if-else if-else
- Lets take a look at some examples structures

for loops

```
for(starting; condition; increment) { stuff to do }
```

Example:

```
char val_in_mem;
int i;
for(i=0; i<0x5FFF; i++) {
    val_in_mem = peekb(0xD000, i);
    printf("The value in mem is: %x \n", val_in_mem);
}
```

while loops

```
while(logical expression) { do_stuff; }
```

Example 1:

```
int i = 0;
while (i < 0x5FFF ) {
    val_in_mem = peekb(0xD000, i);
    i++;
}
```

do while loops

```
do {  
    do_stuff;  
} while(logical expression);
```

Example 1:

```
int i = 0;  
do {  
    val_in_mem = peekb(0xD000, i);  
    i++;  
} while (i<0x5FFF);
```

if constructs

```
if(logical condition) { stuff to do }
```

Example 1:

```
if(a == b)
    { do something; }
```

Example 2:

```
if(a == b)
    { do one thing; }
else if (a == c)
    { do something different; }
else
    { kill the user; }
```

switch statements

```
scanf("%i",&choice) // Get a choice from the user
switch(choice)
{
    case '1':
        print_menu();
        break;
    case '2':
        write_to_mem();
        break;
    default:
        printf("Invalid Selection \n");
}
```

Basic C Constructs

➤ functions

Basic Structure:

```
return_type function_name(parameters) {  
    // the stuff the functions needs to do  
}
```

Example 1:

```
void print_contents(void) {  
    int i;  
    for(i=0; i<100; i++)  
        { printf("here it is %i \n", Array[i]); }  
}
```

Functions Continued

Example 2:

```
int get_first_element(int *A) {  
    int i = 0;  
    return A[i];  
}
```

Example 3:

```
float put(int *A, char start, char end, float k) {  
    int ret_val;  
    do_some_stuff;  
    return ret_val; // here is an example of where there is a return  
                    // type mismatch.  
}
```

Borland Turbo C



- The Help Menu – helps you find function definitions and header files
- Setting you the Output Directory – Place where the .exe file goes
- Make sure you save the file each time you you re-run your program
- The name of the .exe file – TC settings may cause problems.