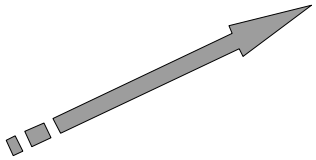
Lecture 7



1. Introduction
2. Binary Representation
3. Hardware and Software
4. High Level Languages
5. Standard input and output
6. Operators, expression and statements
7. Making Decisions
8. Looping
9. Arrays
10. Basics of pointers
11. Strings
12. Basics of functions
13. More about functions
14. Files
14. Data Structures
16. Case study: lottery number generator

Making Decisions

- This lecture is about program "branching", where different actions can occur depending upon the state of data.
- The simplest of which is the conditional expression or query-colon ?:
- C has a conditional operator, ?: which is used in the form

  expr1 ? expr2 : expr3

  boolean expression ? if true : if false
- Remember that in C, non-zero means true, zero means false. If expr1 is true, the value of the expression is that of expr2; if expr1 is false then it is that of expr3

Conditional Expression cont

- Because this is an expression, we can use it on the RHS of an assignment, eg.

  `y=(x>=7)?5:(17-x);`
- So
  - if x is 19, y is 5
  - if x is 2, y is 15
- As the ?: can make code a little difficult to read it is best for short expressions

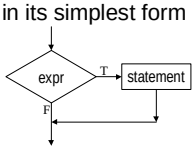
```
/* Example: conditional expressions */

#include <stdio.h>

main()
{
  int n, m, abs, max;

  printf("Enter a positive or negative integer: ");
  scanf("%i", &n);

  printf("\nYou entered %i.\n", n);
  abs = n < 0 ? -n : n;
  printf("Its absolute value is %i.\n", abs);

  printf("\nEnter two integers (e.g. 1 2): ");
  scanf("%i %i", &n, &m);

  printf("\nYou entered %i and %i.\n", n, m);
  max = n > m ? n : m;
  printf("%i is the larger value.\n", max);
}
```

If Statements

- The if statement is the bread and butter of most programmers across many languages
- C provides an if statement, in its simplest form

```
if (expr)
  statement;
```

e.g.
```
if (x==3)
    y++;
```



- if expr evaluates to true (non zero), statement is executed; if not, statement is skipped.

If Statements

- Note that if several statements are to be executed, they must be grouped together to form a block, using braces {}
- e.g.

```
if (x==3){
   y++;
   z*=37;
}
```

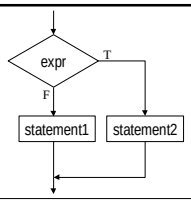
```
/* Example: simple if statement */

#include <stdio.h>

main()
{
  int n;

  printf("Enter an integer: ");
  scanf("%i", &n);

  printf("\nYou entered %i", n);

  if (n >= 0)
    printf(", which isn't negative.\n");
}
```

If Statements

- As with the conditional operator, its possible to have two branches

```
if (expr)
   statement1;
else
   statement2;
```

- Again, statement1 and statement2 can be blocks

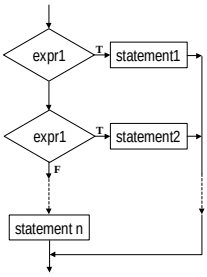



```
/* Example: if...else statement */

#include <stdio.h>

main()
{
  int n;

  printf("Enter an integer: ");
  scanf("%i", &n);

  printf("\nYou entered %i", n);

  if (n >= 0)
    printf(", which isn't
negative.\n");
  else
    printf(", which is negative.\n");
}
```

If Statements

- What if we wish to have a 3-way (or more) choice ?
- We can use a "nested" if…else statement
- Nested basically means one inside another

```
if (expr1)
   statement1;
else
   if (expr2)
      statement2;
      ⋮
   else
      statement n;
```



nestedif.c
if1.bug
if2.bug

```
/* Example: nested if...else statements */

#include <stdio.h>

main()
{
  int n;

  printf("Enter an integer: ");
  scanf("%i", &n);

  printf("\nYou entered %i", n);

  if (n > 0)
    printf(", which is positive.\n");
  else if (n < 0)
    printf(", which is negative.\n");
  else
    printf(", which is neither positive nor negative.\n");
}
```

```
/* BUG ZONE!!!
Example: if...else statement */

#include <stdio.h>

main()
{
  int lucky;

  printf("Enter your lucky number: ");
  scanf("%i", &lucky);

  if (lucky = 7)  /* BUG */
    printf("Your lucky number is %i, same as mine!\n", lucky);
  else
    printf("Your lucky number is %i\n", lucky);
}
```

```
/* BUG ZONE!!!
Example: if statement */

#include <stdio.h>

main()
{
  int dialled = 123;

  printf("You dialled %i\n", dialled);

  if (dialled == 999);  /* BUG */
    puts("Emergency services.");
    puts("Which service do you require?");  /* BUG */
    puts("Police, ambulance, fire or coastguard?");
}
```

Logical Operators

- Consider the following

  "I only attend lectures on Monday at 10 or on Thursday at 10, otherwise I say in bed"

- This could be described by a rather complicated flow diagram involving four "if" boxes:

Logical Operators

- This would make an equally complicated C program. Fortunately C provides three logical operators to get around this.

  && - AND, | | - OR and ! - NOT

  a&&b means both a and b must be true for the result to be true

  a||b means either a or b (or both) must be true for the result to be true

  !a means a must be false for the result to be true

  && has higher precedence than ||

- So we could code it as

```
if (( day==MON || day==TUE ) && time==10)
   go_to_lecture();
else
   stay_in_bed();
```

- Sometimes `if(a==0)` is written `if(!a)`

leapyear.c

```
/* Example: determining leap years */

#include <stdio.h>

main()
{

  /* A leap year is one that is divisible by 4, but not
  divisible by 100. Overriding that, a year divisible
  by 400 is a leap year. */

  int year, leap, div4, div100, div400;

  printf("Enter a year (e.g. 1996): ");
  scanf("%i", &year);

  div4 = year % 4 == 0; /* true if divisible by 4 */
                  /* Could also write  div 4 = !(year % 4) */
  div100 = year % 100 == 0;
  div400 = year % 400 == 0;

  leap = (div4 && !div100) || div400;

  /* Could also write  leap = div4 && !(div100 && !div400); */

  if (leap)
    puts("This is a leap year");
  else
    puts("This is not a leap year");
}
```

Switch Statements

- C provides a statement for multiple choices, based on the value of an integer variable:

```
switch (integer_expression){
     case value1:
       statement(s);
       break;
     case value2:
       statement(s);
       break;
       ⋮
     case value n;
       statement(s);
       break;
     default :
       statement(s);
}
```

- break causes the program to jump to the end of the switch statement

switch.c

```
/* Example: switch statement */

#include <stdio.h>

main()
{
  int d;

  printf("Enter a number from 1 to 9: ");
  scanf("%i", &d);
  putchar('\n');

  switch (d)
  {
    case 1:
      puts("A stitch in time saves nine.");
      break;

    case 2:
    case 6:
    case 9:
      puts("Handsome is as handsome does.");
      break;

    case 3:
      puts("Too many cooks spoil the broth.");
      break;

    case 4:
    case 7:
      puts("Pride comes before a fall.");
      break;

    case 5:
    case 8:
      puts("Many hands make light work.");
      break;

    default:
      puts("Very clever. Try again.");
  }
}
```

switch2.c

```
/* Example: switch statement without breaks */

#include <stdio.h>

main()
{
  int v;

  printf("Enter a number from 1 to 10: ");
  scanf("%i", &v);
  putchar('\n');

  if (v < 1 || v > 10)
  {
    puts("Very clever. I'll use 10.\n");
    v = 10;
  }

  switch (v)
  {
    case 10: puts("Ten...");
    case  9: puts("Nine...");
    case  8: puts("Eight...");
    case  7: puts("Seven...");
    case  6: puts("Six...");
    case  5: puts("Five...");
    case  4: puts("Four...");
    case  3: puts("Three...");
    case  2: puts("Two...");
    case  1: puts("One...");
             puts("ZERO!!");
  }
}
```