MC140: lecture #2

- today's topics:
  - computer basics
  - writing your first program
  - installing LCC
  - creating your program
  - compiling and running your program
  - submitting your program electronically

2/5/01 12:04 1

computer basics, 1:
some drawing instructions.

1. start at bottom of page
2. go straight up for 2cm
3. turn right 45 degrees
4. go straight for 1.4cm
5. turn right 90 degrees
6. go straight for 1.4cm
7. turn right 45 degrees
8. go straight for 2cm
9. turn right 90 degrees
10. go straight for 2cm

2/5/01 12:04 2

computer basics, 2:
commands.

- computer follows commands
  commands = series of instructions
- you will learn how to *command* a computer
  command = program = write instructions
- you understand the commands
- does the computer?
- a question of cognition...
  ⇒ Artificial Intelligence (AI)

2/5/01 12:04 3

computer basics, 3:
components.

- computer = hardware + software
- a computer is organized into logical units:
  (1) input
  (2) output
  (3) memory
  (4) arithmetic and logic (ALU)
  (5) central processing (CPU)
  (6) secondary storage

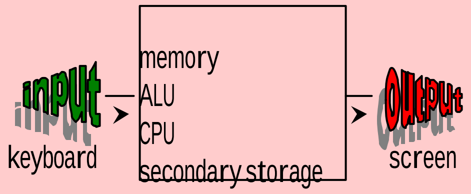
2/5/01 12:04 4

computer basics, 4:
examples of logical units.

- input
  - keyboard, mouse, joystick
- output
  - screen, printer
- memory
  - RAM (random access memory)
- secondary storage
  - hard disk, CDROM, zip disk

2/5/01 12:04 5

computer basics, 5:
a picture.



2/5/01 12:04 6

computer basics, 6:

instructions.

- set of instructions = *program*
- types of instructions:
  - machine language
  - assembly language
  - high-level language (e.g., C)
- program is *compiled* into machine language and then executed
- executing program is called a *job* or *task*

computer basics, 7:

machine language.

- lowest level
- numeric
- computer is comprised of zillions of switches or relays
  - switches = ON or OFF
  - relays = OPEN or CLOSED
- hardware position is abstracted into software as 1's and 0's
- 1's and 0's mean *base 2* = *binary*

computer basics, 8:

assembly language.

- medium level, but still pretty low
- "English" words and abbreviations
- examples:
  - LOAD
  - ADD
  - SHIFT
  - STORE

computer basics, 9:

high-level languages.

- examples: C, BASIC, FORTRAN, Pascal, C++, Java, LISP, Scheme
- even more "English"-like
- high-level languages are *compiled* into machine language

computer basics, 10:

languages example.

- machine language
```
+1300042774
+1400593419
+1200274027
```
- assembly language:
```
LOAD   BASEPAY
ADD    OVERPAY
STORE  GROSSPAY
```
- high-level language:
```
grossPay = basePay + overTimePay;
```

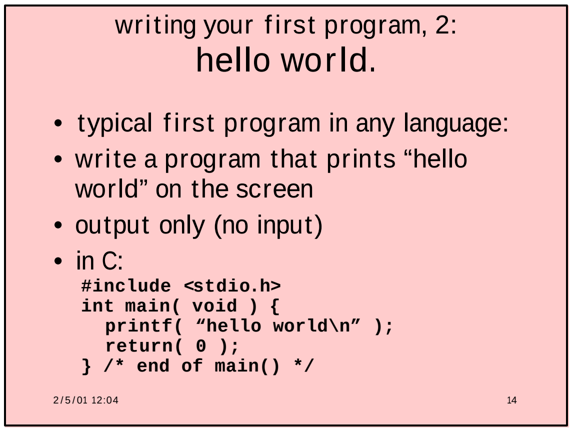
computer basics, 11:

C.

- a program in C consists of modules or *functions*
- there is always a function called *main*
- there also may be:
  - functions from standard C libraries
  - user-defined functions

writing your first program.

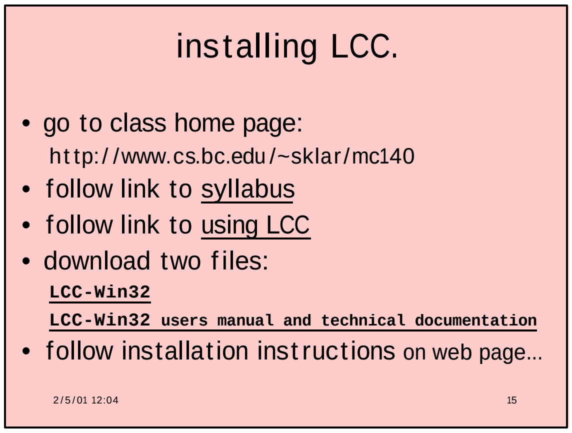
- learning to program is a bit like learning to talk
- first attempts are purely mechanical
- just follow instructions
- gradually you begin to understand what you are doing



writing your first program, 2: hello world.

- typical first program in any language:
- write a program that prints “hello world” on the screen
- output only (no input)
- in C:

```
#include <stdio.h>
int main( void ) {
  printf( “hello world\n” );
  return( 0 );
} /* end of main() */
```



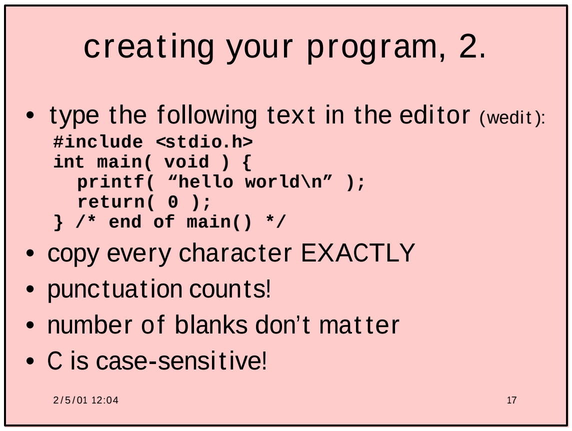
installing LCC.

- go to class home page:
  http://www.cs.bc.edu/~sklar/mc140
- follow link to syllabus
- follow link to using LCC
- download two files:
  `LCC-Win32`
  `LCC-Win32 users manual and technical documentation`
- follow installation instructions on web page...

creating your program.

(detailed instructions on web page)

- create a folder: c:\mc140
- start LCC
- create a new project
- create your source code

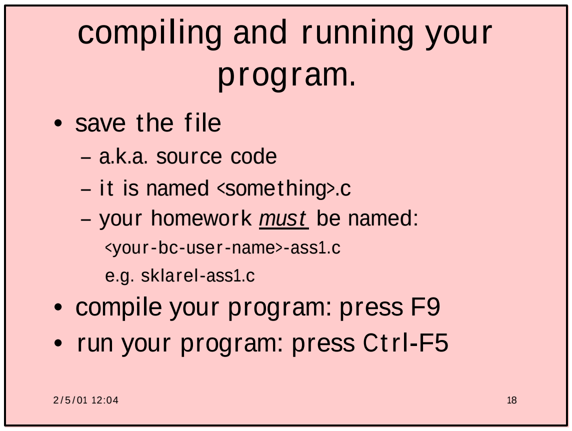


creating your program, 2.

- type the following text in the editor (wedit):

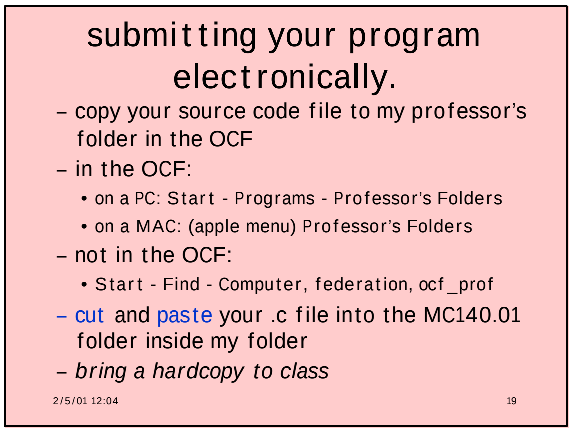
```
#include <stdio.h>
int main( void ) {
  printf( “hello world\n” );
  return( 0 );
} /* end of main() */
```

- copy every character EXACTLY
- punctuation counts!
- number of blanks don’t matter
- C is case-sensitive!



compiling and running your program.

- save the file
  - a.k.a. source code
  - it is named <something>.c
  - your homework *must* be named:
    <your-bc-user-name>-ass1.c
    e.g. sklarel-ass1.c
- compile your program: press F9
- run your program: press Ctrl-F5



submitting your program electronically.

- copy your source code file to my professor's folder in the OCF
- in the OCF:
  - on a PC: Start - Programs - Professor's Folders
  - on a MAC: (apple menu) Professor's Folders
- not in the OCF:
  - Start - Find - Computer, federation, ocf_prof
- cut and paste your .c file into the MC140.01 folder inside my folder
- *bring a hardcopy to class*

2/5/01 12:04 19