CSE 142
Programming I

Style

The Three Ss of Programming

- Syntax
  - ➤Won't compile without proper syntax
- Semantics
  - ➤Won't do what you want without proper semantics
- Style
  - ➤Won't be able to understand the program without proper style

What is Style?

- A program is a document
  - ➤Read by the computer
  - ➤Read by people—not always just you!
- Style is a catch-all term for people-oriented programming
  - ➤comments, spacing, indentation, names
  - ➤clear, straightforward, well-organized code

Why Use Good Style?

- You can read your code, right?
- Can you read it a week/month/year from now?
- Can your TA read it?
  - ➤Can your TA grade it?

Style in CSE 142

- It is common for employers to have style rules for employees
- These rules are often *very* strict
- We won't be so strict, but we will expect you to follow some general guidelines
- We'll show you style points as we go along, starting today

/* Comments */

Comment block at front of program

```
/*************************************************************
 * Program:   Mi_To_Km
 * Purpose:   Miles to kilometers conversion
 * Author:    A. Hacker, 1/18/00  Section AF (Turing)
 *************************************************************/
```

Comment block per major section

```
/* Calculate volume of cylinder and ...
 * Inputs:      radius, height, ...
 * Output:      volume, ...
 * Assumes:     radius, height nonnegative */
```

small ones throughout

```
/* Tell user it's negative. */
```

Comments

- Say *why*, not *what:*
  - NO:

```
/* subtract one from sheep */
sheep = sheep - 1;
```

  - YES:

```
/* account for the sheep that
   the big bad wolf just ate */
sheep = sheep - 1;
```

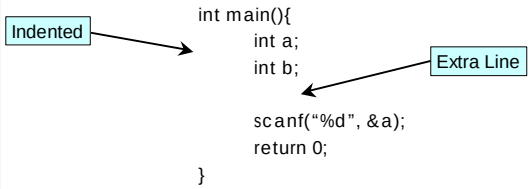

Layout Within a Line

- Leave spaces around operators
  - NO: `y=slope*x+intercept;`
  - YES: `y = slope * x + intercept;`
- Even Better: Use extra parenthesis
  - YES: `y = (slope * x) + intercept;`
- Use spaces to vertically align things:

```
int      foo;
double   bar;
```


Laying Out Lines

- Use blank lines to separate major sections
- Indent subordinate parts



Identifiers

- Identifiers name variables (and more)
  - Letters, digits, and underscores (_)
  - Cannot begin with a digit
  - No reserved words
- Case sensitive
- Don't use all capital letters except for constants (which I'll explain in a moment)

What's in a Name?

- Identifiers are part of the documentation
- Microsoft Excel has over 65,000 variables.
- How long is just right?
  - c
  - cheesecakes
  - numCheesecakes
  - numberOfCheesecakesThatIAteForBreakfastLastWednesday

More on Identifiers

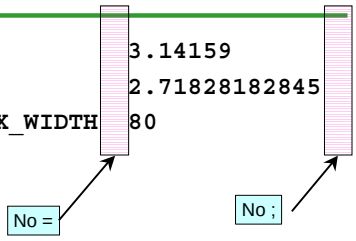
- Avoid similar names
  - mph, Mph, mqh
- Keep simple naming conventions
  - firstNumber
  - first_number

Constants

- Constants are values that do not change
  - ➤pi is always 3.1459…
  - ➤e is always 2.718281828459045…
- Rather than always typing them, we make names for them using **`#define`**
- Make these names upper case!

#define

```
#define PI          3.14159
#define E           2.71828182845
#define BOX_WIDTH   80
.
.
.
area = PI * radius * radius;
```



Why Use Constants?

- Centralize changes
- No “magic numbers”
- Avoid typing errors
- Avoid accidental assignment to constants


```
/* Convert miles per hour to feet per second
 * Author: ...
 * Date: ...
 */

#include <stdio.h>

/* conversion constants. */
#define FEET_PER_MILE     5280.0
#define SECONDS_PER_HOUR  3600

int  main(void)
{
 double miles_per_hour;   /* input mph */
 double feet_per_second;  /* corresponding feet/sec */
 double feet_per_hour;    /* corresponding feet/hr    */

 /* prompt user for input */
 printf("Enter a number of miles per hour: ");
 scanf("%lf", &miles_per_hour);

 /* convert from miles per hour to feet per second */
 feet_per_hour  =  miles_per_hour  *  FEET_PER_MILE;
 feet_per_second  = feet_per_hour  /  SECONDS_PER_HOUR;

 /* format and print results */
 printf("%f miles per hour is equal to %f feet per "
        "second.\n", miles_per_hour, feet_per_second);

 return(0);
}
```


If You Ignore Style...

```
#include<stdio.h>
int main(void){double v1,v2,v3,v4,v5;pr\
intf("Enter a number of miles per hour:\
");scanf("%lf",&v1);v5=v1*14.6666667;pr\
intf("%f miles per hour is equal to %f \
feet per second.\n",v1,v5);return(0);}
```


Summary: What to Do

- Do
  - ➤Use plenty of comments
  - ➤Use white space
  - ➤Use indentation
  - ➤Choose descriptive names
  - ➤Use named constants
  - ➤Use extra parenthesis
  - ➤Be consistent

Summary: What NOT to Do

- Don't
  - Be terse
  - Be tricky
  - Place speed above correctness and simplicity
  - Use magic numbers