# CSE 220 – C Programming

C Fundamentals Part 2

## Administration

- Suggestions about Lab sessions.
- Private Piazza posts can be made public by instructors (unless you specify otherwise).
- Homework #1 is due on Thursday.
  - If you haven't started, you should do so soon
  - You are encouraged to seek help as soon as you need it

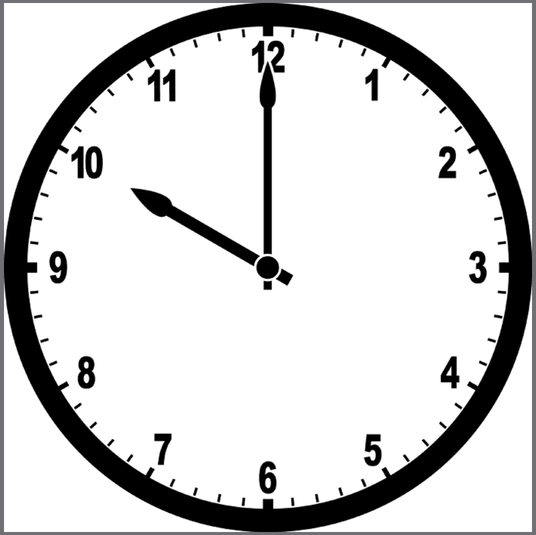
## Before what time of day do homeworks need to be submitted?

Two hours before midnight

10pm

5 hours before the witching hour

This time ➡



http://1.bp.blogspot.com/-cGc6z0lfpqA/TedQM1qkVgI/AAAAAAAADbc/S2oWKNMV1kg/s1600/nclock-10-00_34194_lg.gif

## Outline

- Structure of a C program
- Functions
- Comments
- **Variables**
- Printing output
- Reading input
- Constants
- Identifiers
- Layout

http://www.imd.org/uupload/imd.website/wcc/Top_banners_fundamentals.jpg

## Variables

- Used to store data
- Must have a ***type***:
  - `int`, `float`, `char`, …
- Must be ***declared*** before they can be used***:***
  - Described to the compiler

```
float profit;
int height, width;
```

- Assignment: gives a variable a value:

```
height = 3;
profit = 235.2f; //The f (denoting the number is a float is optional)
```

## Variables

- Example: compute volume of a box

```
int height, width;
height = 3;
width = 2.5;
length = 4;
int length;
volume = height * width * length;
printf("volume is: %d\n", volume);
```

*width is declared as integer, used to store a decimal*

*length is used before it is declared*

- Any errors?

## Variables

- Initialization: gives a variable a default value
- Uninitialized variable:
  - Without a default value
  - Unpredictable result
- Initializer: initial value
- Multiple variables can be initialized in one declaration

```
float rate = 0.65f;

int length;
int width = 1;

int length = 5, width = 2;

int length, width;

int length, width = 1;
int length, width = 1;
```

What value is `length` initialized to?

Unknown

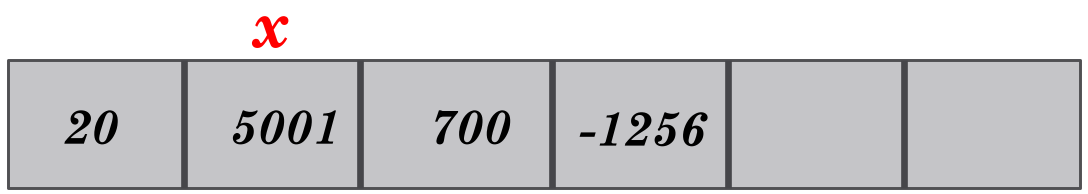
1

0

The same value as `width`

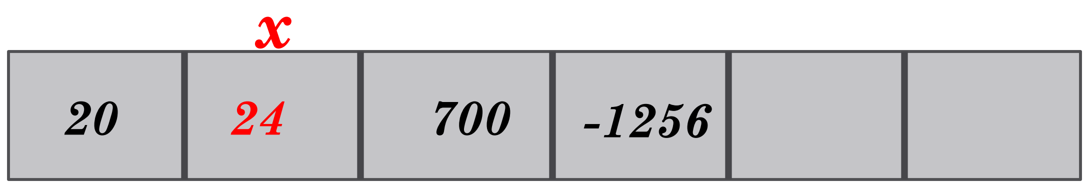
# Variables

- Declare the variable: `int x;`



Representation of Memory

- Initialize the variable: `x = 24;`



- Use the variable: `y = 2 * x;`

## Example

```
/* ****************************************
 * Name: volume.c
 * Purpose: computes volume of a box
 * ****************************************/
#include <stdio.h>

int main(void) {
    int volume, height, length = 5, width = 2;
    volume = height * length * width;
    printf(“Volume: %d\n”, volume);
    return 0;
}
```

Output unpredictable since value stored in variable “height” is unknown

What command will compile the previous program ("volume.c")?

```
gcc volume.c
gcc volume.c -o volume
gcc -o volume volume.c
./volume
```

What command will compile the previous program ("volume.c") into an executable called "volume"?

```
gcc volume.c
```

```
gcc volume.c -o volume
```

```
gcc -o volume volume.c
```

```
./volume
```

What command execute the program called "volume"?

```
gcc volume.c
gcc volume.c -o volume
gcc -o volume volume.c
./volume
```

## Outline

- Structure of a C program
- Functions
- Comments
- Variables
- **Printing output**
- Reading input
- Constants
- Identifiers
- Layout

http://www.imd.org/uupload/imd.website/wcc/Top_banners_fundamentals.jpg

## Printing Strings Revisited

- `printf` function
- Defined in stdio.h
  - Meaning you need the following directive to use it
  - `#include<stdio.h>`
- Prints the value enclosed in quotation marks
- Does not print the quotation marks

```
int x = 2;
printf("The value is x\n");
```

```
The value is x
```

- Can print the value of x

```
int x = 2;
printf("The value is %d\n", x);
```

```
The value is 2
```

- Does not advance to the next line unless told so: `\n`

## How do you print quotation marks (") ?

- Impossible
- `printf("\"")`
- `printf(""")`
- `printf("Quotation Mark")`

## Outline

- Structure of a C program
- Functions
- Comments
- Variables
- Printing output
- **Reading input**
- Constants
- Identifiers
- Layout

http://www.imd.org/uupload/imd.website/wcc/Top_banners_fundamentals.jpg

# Reading Input

- `scanf`: reads the value entered by the user
- `scanf("%d", &x);`
  - reads an integer and stores it in variable x
- `scanf("%f", &y);`
  - reads a float and stores it in variable y
- Should first declare the variables before using them:

```
int x;
float y;
scanf("%d", &x);
scanf("%f", &y);
```

- Why the ampersands (`&`)?
  - I'll explain later (pointers).

## Example

```
int main(void) {
    int volume, height, length, width;

    //Ask for the input and read it
    printf(“Enter the height:\n”);
    scanf(“%d”, &height);
    printf(“Enter the length:\n”);
    scanf(“%d”, &length);
    printf(“Enter the width:\n”);
    scanf(“%d”, &width);

    //Compute the volume and output it
    volume = height * length * width;
    printf(“Volume: %d\n”, volume);
    return 0;
}
```

## Outline

- Structure of a C program
- Functions
- Comments
- Variables
- Printing output
- Reading input
- **Constants**
- Identifiers
- Layout

http://www.imd.org/uupload/imd.website/wcc/Top_banners_fundamentals.jpg

# Names for Constants

```
float area1, area2, area3, radius1,
 radius2, radius3;
float perimeter1, perimeter2,
 perimeter3;
//Initialize variable from user input
…..
//Compute the areas and perimeters
area1 = 3.14*radius1*radius1;
area2 = 3.14*radius2*radius2;
area3 = 3.14*radius3*radius3;
perimeter1 = 2*3.14*radius1;
perimeter2 = 2*3.14*radius2;
perimeter3 = 2*3.14*radius3;
```

What happens when your application requires more accuracy?

# Names for Constants

```
float area1, area2, area3, radius1,
 radius2, radius3;
float perimeter1, perimeter2,
 perimeter3;
//Initialize variable from user input
…..
//Compute the areas and perimeters
area1 = 3.1415*radius1*radius1;
area2 = 3.1415*radius2*radius2;
area3 = 3.1415*radius3*radius3;
perimeter1 = 2*3.1415*radius1;
perimeter2 = 2*3.1415*radius2;
perimeter3 = 2*3.1415*radius3;
```

Change every occurrence of 3.14 with 3.1415

## Names for Constants

- Macro definition: use to name constants

```
#define PI  3.14159f
#define SCALE_FACTOR (5.0f /9.0f)
area = PI*radius*radius;
```

- The preprocessor replaces every occurrence by the value it represents
- If expression contains operators, it should be enclosed by parentheses
- Cannot change value of PI:
  - `PI = 3.1 //results in error`
- Convention: use all capital letters for constant names

## f suffix on float constants

Is `3.4` the same as `3.4f`?

Yes

No, only `3.4` is a float

No, only `3.4f` is a float

No, `3.4` is an integer

## Outline

- Structure of a C program
- Functions
- Comments
- Variables
- Printing output
- Reading input
- Constants
- **Identifiers**
- Layout

http://www.imd.org/uupload/imd.website/wcc/Top_banners_fundamentals.jpg

## Identifiers

- Names for macros, variables, functions
- May contain letters, digits and underscores
- Must begin by letter or underscore
- Case sensitive
- Valid names:
  - `x, y, first_name, lastName, _age, value1, value2, steelboxwidth, steelBoxWidth`
- Invalid names:
  - `1st_value, last-name, scale factor`
- Conventions:
  - Lower case, separate by underscore: `box_height, box_width`
  - Lower case, separate by uppercase: `boxHeight, boxWidth`

## Which names are valid identifiers?

```
my_ferret_is_sleepy

tHiSIsAvAlidNaMe

one_two_three_4

_________
```

## Keywords

- Special words in C
- Cannot be used as identifiers

|  |  |  |  |  |
|---|---|---|---|---|
| auto | break | case | char | const |
| continue | default | do | double | else |
| enum | extern | float | for | goto |
| if | int | long | register | return |
| short | signed | sizeof | static | struct |
| switch | typedef | union | unsigned | void |
| volatile | while |  |  |  |
| inline | restrict | _Bool | _Complex | _Imaginary |

## Outline

- Structure of a C program
- Functions
- Comments
- Variables
- Printing output
- Reading input
- Constants
- Identifiers
- **Layout**

http://www.imd.org/uupload/imd.website/wcc/Top_banners_fundamentals.jpg

# Layout of a C program

- Readability is important
- Indent your program
- Add blank lines
- Divide long statements into multiple lines

```
printf(“The volume of the box in cubic feet is %d”,
            height*width*length);
```

- Cannot add spaces in the middle of a token

```
printf(“The volume of the box in cubic
            feet is %d”, height*width*length);
/* wrong */
```

## Example

Hard to understand formatting:

```
int main(void) {
int a, h, b;printf(“Enter the height:\n”);scanf(“%d”,
&h);printf(“Enter the length:\n”);scanf(“%d”,
&b);a=h*b;printf(“Area: %d\n”, a);return 0;}
```

However, this compiles and is logically correct.

Easy to understand formatting:

```
int main(void) {
    int area, height, length;

    /* Read values from user */
    printf(“Enter the height:\n”);
    scanf(“%d”, &height);
    printf(“Enter the length:\n”);
    scanf(“%d”, &length);

    /* Compute the area and print it */
    area = height * length;
    printf(“Area: %d\n”, area);

   return 0;
}
```

## How many lines are printed?

```
printf("This is line one");
printf("This is line two");
printf("This is line three");
```

0

1

3

Trick Question (-1)

## How many lines are printed?

```
printf("This is line one\n");
printf("This is line two\n");
printf("This is line three\n");
```

0

1

3

Trick Question (-1)

## How many lines are printed?

```
printf("This is line one\n"
       "This is line two\n"
       "This is line three\n");
```

0

1

3

Trick Question (-1)