CSCI 2132
Software Development

Lecture 14:

Software Development Life Cycle

Instructor: Vlado Keselj

Faculty of Computer Science

Dalhousie University

Previous Lecture

- *(Fire Alarm)*
- Integer and floating-point representation
- Character types

Reading Characters

- scanf can be used to read a character, but it does not ignore white space
- For example, the following statements are not equivalent. (Find an input example.)

```
scanf("%c", &ch);
scanf(" %c", &ch);
```

- `getchar` and `putchar` are used for input and output at the character level; e.g.:

```
int ch = getchar();
putchar(ch);
```

- `int` is used as return to detect EOF

Code Example

```
#include <stdio.h>

int main() {
  int ch;
  while (EOF != (ch = getchar()) && ch != ’\n’)
    if (’a’ <= ch && ch <= ’z’)
      ch = ch - ’a’ + ’A’;
      putchar(ch);
    }
  putchar(’\n’);
  return 0;
}
```

Code Example

- Consider:

```
EOF != (ch = getchar()) && ch != ’\n’
```

- Are brackets necessary?

Type Conversions

- Consider the following code:

```
float f = 3.4;
```

- Works in C (not in Java) due to implicit type conversion
- Implicit type conversion takes place in:
  - operands, and
  - assignments

Example with Operands

- Example

```
float f;
double d;
int i;
d = d + f;
f = f + i;
```

- f is promoted to double, and i is promoted to double
- operands are promoted to ‘narrowest’ type which will accomodate both

Implicit Conversion in Assignment

- Example:

  ```
  int i = 8.92;
  ```

- The right side is converted to the type of the left side

Type Casting

- Type Casting, or explicit type conversion
- Syntax: `(type) expression`
- Example

```
float f, frac_part;
frac_part = f - (int)f;
```

- What is calculated in `frac_part`?

Another Example

```
float quotient;
int dividend = 5;
int divisor = 4;
quotient = dividend / divisor;
quotient = (float) dividend / divisor;
quotient = (float) (dividend / divisor);
quotient = 1.0f * dividend / divisor;
```

- What values are assigned in these four statements to `quotient`?

Type Definitions Using typedef

- We can define types using:

```
typedef typename alternative_name
```

- One example:

```
typedef int Bool;
Bool flag;
```

- Particularly useful with more complex types, as we will see later

The sizeof Operator

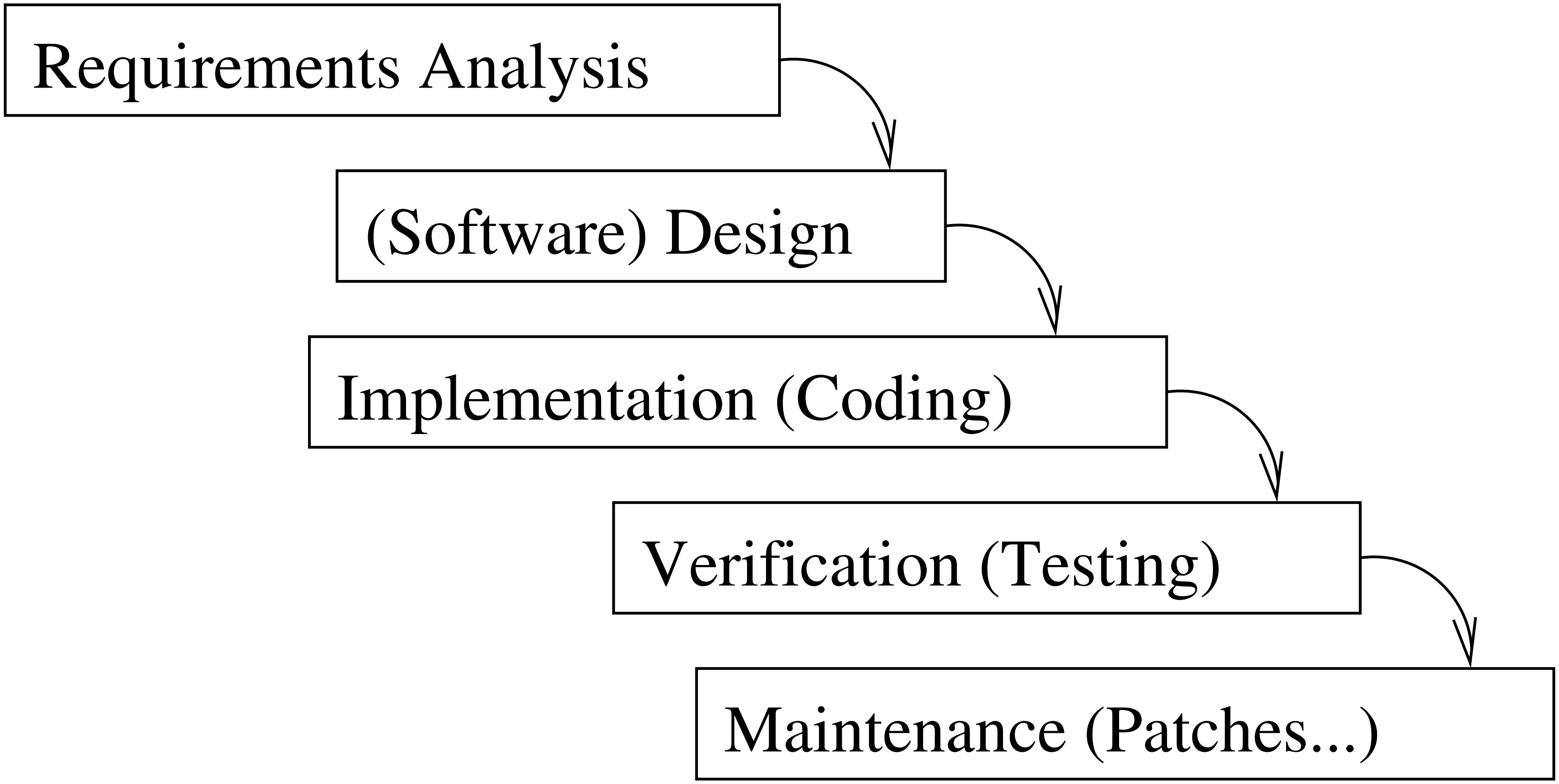
- Size of many C types are implementation-defined
- `sizeof` operator = number of bytes required to store a type
- Syntax: `sizeof(`*type*`)`
- Example: `sizeof(char)`
- `sizeof` can also be applied to variables; e.g.:
  `int i; printf("%d\n", sizeof(i));`

Software Development Life Cycle (SDLC)

- SDLC is a general term that describes structure imposed on the development of a software product
- Purpose
  - To reduce the risk of missing the deadline
  - To ensure product quality
  - To prevent “scope creep”, etc.
- Many models have been proposed to describe SDLC

The Waterfall Model

- A sequential design process



Requirements Analysis

(Software) Design

Implementation (Coding)

Verification (Testing)

Maintenance (Patches...)

Waterfall: Advantages and Disadvantages

- Advantages
  - Natural and easy to understand
  - Widely used
  - Reinforces notion of “design before coding”
  - Clear milestones
- Disadvantages
  - Often not practical
  - Clients may change requirements
  - Designers may not be aware of implementation difficulties

The Rapid Prototyping Model

1. Gathering preliminary requirements
2. Fast prototyping
3. User evaluation of the prototype
4. Repeat the above steps if necessary
5. Discard the prototype and develop the software using a formal process

Rapid Prototyping: Advantages and Disadvantages

- Advantages
  - Ensure that software product meets client's requirements
  - Reduce time and cost if client requests changes during the process
- Disadvantages
  - Adequate and appropriate user involvement may not always be possible
  - Cost of prototype development

More about Models

- There are many other models
- To be studied in the Software Engineering course
- Choose an appropriate model depending on the particular software to be developed