



**Shree H. N. Shukla Institute of Pharmaceutical  
Education and Research, Rajkot**

**B. Pharm  
Semester-2**

**STUDY MATERIAL**

**Subject Name: computer application in pharmacy**

**Subject Code: BP204TP**

## CHAPTER 1 :

**Number system, division Concept of Information Systems and Software****Contains**

Number system: Binary number system, Decimal number system, Octal number system, Hexadecimal number systems, conversion decimal to binary, binary to decimal, octal to binary etc, binary addition, binary subtraction – One's complement, Two's complement method  
Concept of Information Systems and Software : Information gathering, , data flow diagrams, process specifications, input/output design

## Number system

The technique to represent and work with numbers is called **number system**. **Decimal number system** is the most common number system. Other popular number systems include **binary number system**, **octal number system**, **hexadecimal number system**, etc.

## Decimal Number System

Decimal number system is a **base 10** number system having 10 digits from 0 to 9. This means that any numerical quantity can be represented using these 10 digits. Decimal number system is also a **positional value system**. This means that the value of digits will depend on its position. Let us take an example to understand this.

Say we have three numbers – 734, 971 and 207. The value of 7 in all three numbers is different–

- In 734, value of 7 is 7 hundreds or 700 or  $7 \times 100$  or  $7 \times 10^2$
- In 971, value of 7 is 7 tens or 70 or  $7 \times 10$  or  $7 \times 10^1$
- In 207, value of 7 is 7 units or 7 or  $7 \times 1$  or  $7 \times 10^0$

The weightage of each position can be represented as follows –

$10^5$	$10^4$	$10^3$	$10^2$	$10^1$	$10^0$
--------	--------	--------	--------	--------	--------

In digital systems, instructions are given through electric signals; variation is done by varying the voltage of the signal. Having 10 different voltages to implement decimal number system in digital equipment is difficult. So, many number systems that are easier to implement digitally have been developed. Let's look at them in detail.

## Binary Number System



**Octal number system** has eight digits – 0, 1, 2, 3, 4, 5, 6 and 7. Octal number system is also a positional value system with where each digit has its value expressed in powers of 8, as shown here –

$8^5$	$8^4$	$8^3$	$8^2$	$8^1$	$8^0$
-------	-------	-------	-------	-------	-------

Decimal equivalent of any octal number is sum of product of each digit with its positional value.

$$726_8 = 7 \times 8^2 + 2 \times 8^1 + 6 \times 8^0$$

$$= 448 + 16 + 6$$

$$= 470_{10}$$

## Hexadecimal Number System

**Octal number system** has 16 symbols – 0 to 9 and A to F where A is equal to 10, B is equal to 11 and so on till F. Hexadecimal number system is also a positional value system with where each digit has its value expressed in powers of 16, as shown here –

$16^5$	$16^4$	$16^3$	$16^2$	$16^1$	$16^0$
--------	--------	--------	--------	--------	--------

Decimal equivalent of any hexadecimal number is sum of product of each digit with its positional value.

$$27FB_{16} = 2 \times 16^3 + 7 \times 16^2 + 15 \times 16^1 + 10 \times 16^0$$

$$= 8192 + 1792 + 240 + 10$$

$$= 10234_{10}$$

## Number System Relationship

The following table depicts the relationship between decimal, binary, octal and hexadecimal number systems.

HEXADECIMAL	DECIMAL	OCTAL	BINARY
0	0	0	0000
1	1	1	0001

---

2	2	2	0010
3	3	3	0011
4	4	4	0100
5	5	5	0101
6	6	6	0110
7	7	7	0111
8	8	10	1000
9	9	11	1001
A	10	12	1010
B	11	13	1011
C	12	14	1100
D	13	15	1101
E	14	16	1110
F	15	17	1111

Complements are used in digital computers in order to simplify the subtraction operation and for the logical manipulations. For the Binary number (base-2) system, there are two types of complements: 1's complement and 2's complement.

### 1's Complement of a Binary Number

There is a simple algorithm to convert a binary number into 1's complement. To get 1's complement of a binary number, simply invert the given number.

### 2's Complement of a Binary Number

There is a simple algorithm to convert a binary number into 2's complement. To get 2's complement of a binary number, simply invert the given number and add 1 to the least significant bit (LSB) of given result.

### Differences between 1's complement and 2's complement

These differences are given as following below –

1's complement	2's complement
To get 1's complement of a binary number, simply invert the given number.	To get 2's complement of a binary number, simply invert the given number and add 1 to the least significant bit (LSB) of given result.
1's complement of binary number 110010 is 001101	2's complement of binary number 110010 is 001110
Simple implementation which uses only NOT gates for each input bit.	Uses NOT gate along with full adder for each input bit.
Can be used for signed binary number representation but not suitable as ambiguous representation for number 0.	Can be used for signed binary number representation and most suitable as unambiguous representation for all numbers.
0 has two different representation one is -0 (e.g., 1111 in five bit register) and second is +0 (e.g., 0000 in five bit register).	0 has only one representation for -0 and +0 (e.g., 0000 in five bit register). Zero (0) is considered as always positive (sign bit is 0)
For k bits register, positive largest number that can be stored is $(2^{(k-1)}-1)$ and negative lowest number	For k bits register, positive largest number that can be stored is $(2^{(k-1)}-1)$ and negative lowest number that

that can be stored is $-(2^{(k-1)}-1)$ .	can be stored is $-(2^{(k-1)})$ .
<i>end-around-carry-bit</i> addition occurs in 1's complement arithmetic operations. It added to the LSB of result.	<i>end-around-carry-bit</i> addition does not occur in 2's complement arithmetic operations. It is ignored.
1's complement arithmetic operations are not easier than 2's complement because of addition of <i>end-around-carry-bit</i> .	2's complement arithmetic operations are much easier than 1's complement because of there is no addition of <i>end-around-carry-bit</i> .
Sign extension is used for converting a signed integer from one size to another.	Sign extension is used for converting a signed integer from one size to another.

What's information gathering?

When it comes to getting a clear information gathering concept, the simplest way to define it would be the process of collecting information about something you are interested in.

For those in the cybersecurity industry, this is the first step to take during the earlier stages of any hacking activity (both [cracking](#) and [ethical hacking](#)), when any black- or white-hat researcher needs to gain as much information as possible about the desired target.

While it's a fun activity for some researchers, information gathering is also one of the most time-consuming tasks during the intel-recon process, and that is why time management is so important.

What are the objectives of information gathering in cybersecurity?

Any basic cybersecurity information gathering process often includes these two types of data collection goals:

1. Collecting network data: Such as public, private and associated domain names, network hosts, public and private IP blocks, routing tables, TCP and UDP running services, SSL certificates, [open ports](#) and more.

2. Collecting system-related information: This includes user enumeration, system groups, OS hostnames, OS system type (probably by [fingerprinting](#)), system banners (as seen in the [banner grabbing](#) blog post), etc.

But there's a lot more involved. Let's learn about it, by exploring the most popular techniques used during this phase.

### Information gathering techniques

Ethical hackers use a big variety of techniques and tools to get this precious information about their targets, as well as locations and data collection software they'll be using towards the information gathering goal.

Let's look at the top methods used to gather information about any target.

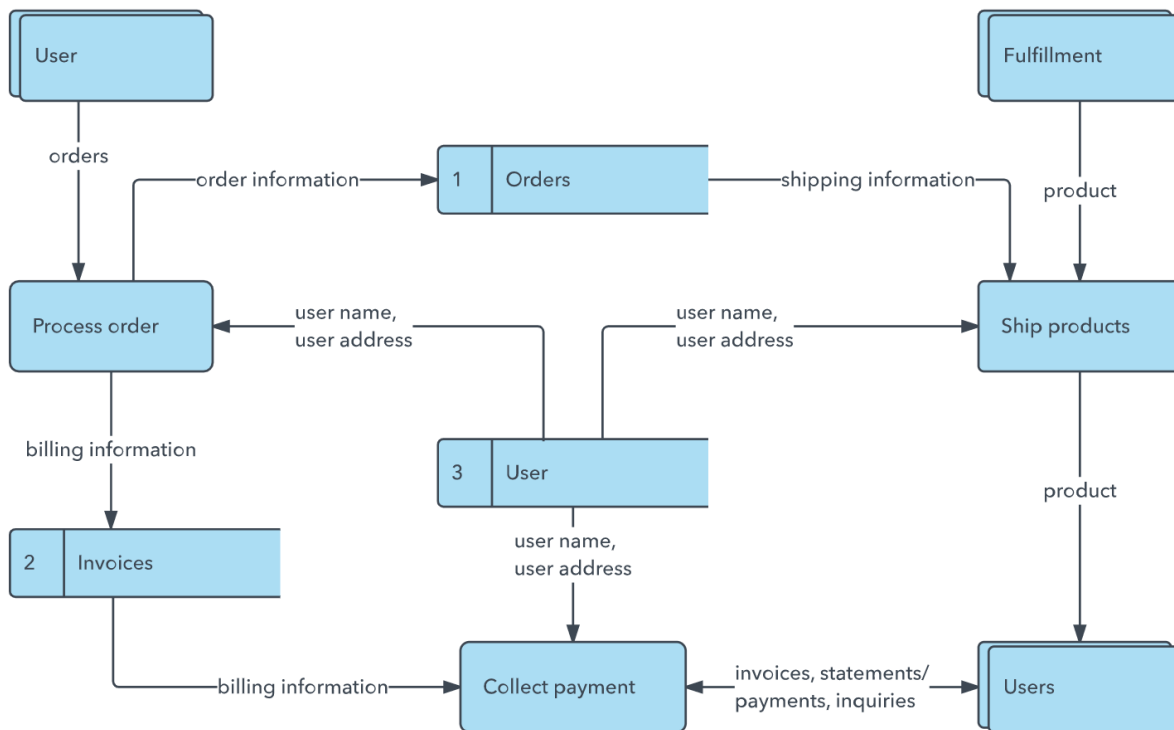
### How to gather information?

- [Social engineering](#): This includes in-person chat, phone conversations and email spoofing attacks. What all these methods have in common is the psychology of human weakness, needed to get maximum data about the target.
- Search engines: Web crawlers can be used to fetch information about anything, and this includes companies, persons, services, and even real hacks, as seen in our previous article about [Google Hacking](#).
- Social networks: Facebook, Twitter, LinkedIn and other social networks are great sources of information to build a profile, especially when targeting individuals.
- Domain names: These are registered by organizations, governments, public and private agencies, and people. Therefore, they're a great starting point when you want to investigate someone. Personal information, associated domains, projects, services and technologies can be found by inspecting domain name information.
- Internet servers: authoritative DNS servers are a great source of information, as they often include every single surface point exposed to the Internet—which means a direct link to related services such as HTTP, email, etc. In our previous article about [passive DNS](#), we analyzed the importance of DNS servers, and especially passive DNS-recon services, such as the ones we offer here at SecurityTrails.

### What is a data flow diagram?



A data flow diagram (DFD) maps out the flow of information for any process or system. It uses defined symbols like rectangles, circles and arrows, plus short text labels, to show data inputs, outputs, storage points and the routes between each destination. Data flowcharts can range from simple, even hand-drawn process overviews, to in-depth, multi-level DFDs that dig progressively deeper into how the data is handled. They can be used to analyze an existing system or model a new one. Like all the best diagrams and charts, a DFD can often visually “say” things that would be hard to explain in words, and they work for both technical and nontechnical audiences, from developer to CEO. That’s why DFDs remain so popular after all these years. While they work well for data flow software and systems, they are less applicable nowadays to visualizing interactive, real-time or database-oriented software or systems.



## History of the DFD

Data flow diagrams were popularized in the late 1970s, arising from the book , by computing pioneers Ed Yourdon and Larry Constantine. They based it on the “data flow graph” computation models by David Martin and Gerald Estrin. The structured design concept took off in the software engineering field, and the DFD method took off with it. It became more popular in business circles, as it was applied to business analysis, than in academic circles.

Also contributing were two related concepts:

- Object Oriented Analysis and Design (OOAD), put forth by Yourdon and Peter Coad to analyze and design an application or system.
- Structured Systems Analysis and Design Method (SSADM), a waterfall method to analyze and design information systems. This rigorous documentation approach contrasts with modern agile approaches such as Scrum and Dynamic Systems Development Method (DSDM.)

Three other experts contributing to this rise in DFD methodology were Tom DeMarco, Chris Gane and Trish Sarson. They teamed up in different combinations to be the main definers of the symbols and notations used for a data flow diagram.



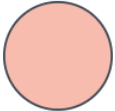





## Symbols and Notations Used in DFDs

Two common systems of symbols are named after their creators:

- Yourdon and Coad
- Yourdon and DeMarco
- Gane and Sarson

One main difference in their symbols is that Yourdon-Coad and Yourdon-DeMarco use circles for processes, while Gane and Sarson use rectangles with rounded corners, sometimes called lozenges. There are other symbol variations in use as well, so the important thing to keep in mind is to be clear and consistent in the shapes and notations you use to communicate and collaborate with others.

---

Notation	Yourdon and Coad	Gane and Sarson
External Entity		
Process		
Data Store		
Data Flow		

Using any convention's DFD rules or guidelines, the symbols depict the four components of data flow diagrams.

1. **External entity:** an outside system that sends or receives data, communicating with the system being diagrammed. They are the sources and destinations of information entering or leaving the system. They might be an outside organization or person, a computer system or a business system. They are also known as terminators, sources and sinks or actors. They are typically drawn on the edges of the diagram.
2. **Process:** any process that changes the data, producing an output. It might perform computations, or sort data based on logic, or direct the data flow based on business rules. A short label is used to describe the process, such as "Submit payment."

3. **Data store:** files or repositories that hold information for later use, such as a database table or a membership form. Each data store receives a simple label, such as “Orders.”
4. **Data flow:** the route that data takes between the external entities, processes and data stores. It portrays the interface between the other components and is shown with arrows, typically labeled with a short data name, like “Billing details.”

Want more detail? [Here](#) is a comprehensive look at diagram symbols and notations and how they're used.

## DFD rules and tips

- Each process should have at least one input and an output.
- Each data store should have at least one data flow in and one data flow out.
- Data stored in a system must go through a process.
- All processes in a DFD go to another process or a data store.