FUNDAMENTALS OF COMPUTING & COMPUTER PROGRAMMING

UNIT – I – 16 MARKS

1. Define computer. Explain the characteristics briefly? (MAY 2009\FEB 2009)

A computer is a programmable machine or device that performs pre-defined or programmed computations or controls operations that are expressible in numerical or logical terms at high speed and with great accuracy.

Characteristics of Computers

· Speed

· Accuracy.

· Automation.

· Endurance.

· Versatility.

· Storage.

· Cost Reduction.

---

2. With suitable examples, explain about Number systems. (JAN 2009)

A number system is a set of rules and symbols used to represent a number. There are several different number systems. Some examples of number systems are as follows:

· Binary (base 2)

· Octal (base 8)

· Decimal (base 10)

· Hexadecimal (base 16)

Decimal and Hexadecimal numbers can each be represented using binary values. This enables decimal, hexadecimal, and other number systems to be represented on a computer which is based around binary (0 or 1 / off or on). The base (or radix) of a number system is the number of units that is equivalent to a single unit in the next higher counting space. In the decimal number system, the symbols 0-9 are used in combination to represent a number of any sizes.

For example, the number 423 can be viewed as the following string of calculations: (4 x 100) + (2 x 10) + (3 x 1) = 400 + 20 + 3 = 423

---

3. Describe evolution of computer? (JAN 2009 / MAY 2009)

· Abacus

· Astrolabe

· Pascaline

· Stepped Reckoner

· Difference Engine

· Analytical Engine

· Punch Cards

· ENIAC (Electrical Numerical Integrator and Calculator)

· Von Neumann Machine

---

4. Explain various generations of computers with features? (FEB 2009/FEB 2010)

Generation of Computers

Each phase of computer development is known as a separate generation of computers. The computer can be classified into four generations according to their type of electronic circuits such as vacuum tube, transistor, IC etc.

(a) The First Generation Computers (1949-55)

Main Features:

1) The computers of this generation used vacuum tubes.

2) These computers used machine language for giving instructions.

3) They used the concept of stored program.

4) These computers were 5000 times faster than the MARK-I.

5) The first generation computers were welcomed by Government and Universities.

Limitations:

1) These computers were very big in size. The ENIAC machine was 30 x 50 feet in size and

30 tons in weight. So, these machines required very large space for their workings.

2) Their power consumption was very high.

3) These computers had slow operating speed and small computing capacity.

4) These computers had a very small memory.

(b) The Second Generation Computers (1956-65) Main Features:

1) The computers of this generation replaced vacuum tubes with transistors.

2) Magnetic cores were invented for storage.

3) Different magnetic storage devices were developed in this generation.

4) Commercial applications were developed during this period. Eighty percent of these computers were used in business and industries.

(c) Third Generation Computers (1966-75) Main Features:

· The third generation computers replaced transistors with' Integrated Circuits'. These

Integrated Circuits are also known as chips.

· The size of main memory was increased and reached about 4 megabytes.

· Magnetic disk technology had been improved and drive having capacity upto 100

MBPS came into existence.

· The CPU becomes more powerful with the capacity of carrying out 1 million instructions per second.

· This generation computers were relatively inexpensive and faster.

· The application area also increased in this generation. The computers were used in other areas like education, small businesses survey, analysis along with their previous usage areas.

(d) The Fourth Generation Computers (1976-Present) Main Features:

i.The fourth generation computers replaced small scale integrated circuits and medium scale integrated circuits with the microprocessors chip.

ii. Semiconductor memories replaced magnetic core memories.

iii. The hard-disks are available of the sizes upto 200 GB. The RAID technology

(Redundant Array of Inexpensive Disks) gives storage upto thousands of GB. iv. Computer cost came down rapidly in this generation.

v. Application of computers is increased in various areas like visualization, parallel computing, multimedia etc.

(e) The Fifth Generation Computers

Mankind along with the advancement in science and technology is working hard to bring the Vth Generation of computer. These computers will have the capability of thinking on their own like an man with the help of Artificial Intelligence (AI). the 21st century will be better, faster, smaller and smarter computers.

---

5. Explain the fundamental units of a computer with a block diagram? (Or)

Explain the basic computer organization in detail? (JAN 2009\MAY 2009)

A computer can process data, pictures, sound and graphics. They can solve highly complicated problems quickly and accurately.

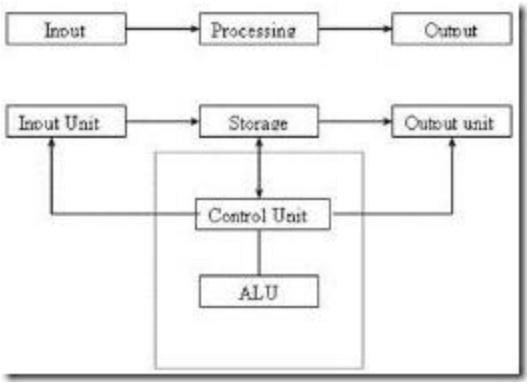
Input Unit:

Computers need to receive data and instruction in order to solve any problem. Therefore we need to input the data and instructions into the computers. The input unit consists of one or more input devices. Keyboard is the one of the most commonly used input device. Other commonly used input devices are the mouse, floppy disk drive, magnetic tape, etc. All the input devices perform the following functions.

· Accept the data and instructions from the outside world.

· Convert it to a form that the computer can understand.

· Supply the converted data to the computer system for further processing.



Storage Unit:

Block Diagram of Computer

The storage unit of the computer holds data and instructions that are entered through the input unit, before they are processed. It preserves the intermediate and final results before these are sent to the output devices. It also saves the data for the later use.

Types of Storage Devices:

1. Primary Storage:

1. Stores and provides very fast.

2. This memory is generally used to hold the program being currently executed in the computer, the data being received from the input unit, the intermediate and final results of the program.

3. The primary memory is temporary in nature. The data is lost, when the computer is switched off.

4. In order to store the data permanently, the data has to be transferred to the secondary memory. The cost of the primary storage is more compared to the secondary storage.

2. Secondary Storage:

1. It stores several programs, documents, data bases etc.

2. The programs that run on the computer are first transferred to the primary memory before it is actually run.

3. Whenever the results are saved, again they get stored in the secondary memory.

4. The secondary memory is slower and cheaper than the primary memory. Some of the commonly used secondary memory devices are Hard disk, CD, etc.,

Memory Size:

All digital computers use the binary system, i.e. 0's and 1's. Each character or a number is represented by an 8 bit code. The set of 8 bits is called a byte. A Character occupies 1 byte space. A numeric occupies 2 byte space. Byte is the space occupied in the memory. The size of the primary storage is specified in KB (Kilobytes) or MB (Megabyte). One KB is equal to 1024 bytes and one MB is equal to 1000KB. The size of the primary storage in a typical PC usually starts at 16MB. PCs having 32 MB, 48MB, 128 MB, 256MB memory are quite common.

Output Unit:

The output unit of a computer provides the information and results of a computation to

outside world. Printers, Visual Display Unit (VDU) are the commonly used output devices. Other commonly used output devices are floppy disk drive, hard disk drive, and magnetic tape drive. **Arithmetic Logical Unit:**

All calculations are performed in the Arithmetic Logic Unit (ALU) of the computer. It also does comparison and takes decision. The ALU can perform basic operations such as addition, subtraction, multiplication, division, etc and does logic operations viz, >, <, =, 'etc. Whenever calculations are required, the control unit transfers the data from storage unit to ALU once the computations are done, the results are transferred to the storage unit by the control unit and then it is send to the output unit for displaying results.

Control Unit:

It controls all other units in the computer. The control unit instructs the input unit, where to store the data after receiving it from the user. It controls the flow of data and instructions from the storage unit to ALU. It also controls the flow of results from the ALU to the storage unit. The control unit is generally referred as the central nervous system of the computer that control and synchronizes its working.

Central Processing Unit:

The control unit and ALU of the computer are together known as the Central Processing Unit

(CPU). The CPU is like brain performs the following functions:

• It performs all calculations.

• It takes all decisions.

• It controls all units of the computer.

A PC may have CPU-IC such as Intel 8088, 80286, 80386, 80486, Celeron, Pentium, Pentium Pro, Pentium II, Pentium III, Pentium IV, Dual Core, and AMD etc.

6. Explain the classification of computers? (MAY 2009\FEB 2009\FEB 2010)

Personal Computers:

CLASSIFICATION OF COMPUTERS

A personal computer (PC) is a self-contained computer capable of input, processing, output, and storage. A personal computer is designed to be a single-user computer and must have at least one input device, one output device, a processor, and memory. The three major groups of PCs are desktop computers, portable computers, and handheld computers. Desktop Computers: A desktop computer is a PC designed to allow the system unit, input devices, output devices, and other connected devices to fit on top of, beside, or under a user's desk or table. This type of computer may be used in the home, a home office, a library, or a corporate setting.

Portable Computers:

A portable computer is a PC small enough to be moved around easily. As the name suggests, a laptop computer fits comfortably on the lap. As laptop computers have decreased in size, this type of computer is now more commonly referred to as a notebook computer. Manufacturers recently began introducing a new type of computer called the tablet PC, which has a liquid crystal display (LCD) screen on which the user can write using a special-purpose pen, or stylus. Tablet PCs rely on digital ink technology that allows the user to write on the screen. Another type of portable computer, called a wearable computer, is worn somewhere on the body, thereby providing a user with access to mobile computing capabilities and information via the Internet.

Handheld Computers:

An even smaller type of personal computer that can fit into the hand is known as a handheld computer (also called simply handheld, pocket PC, or Palmtop). In recent years, a type of handheld computer called a personal digital assistant (PDA) has become widely used for performing calculations, keeping track of schedules, making appointments, and writing memos. Some handheld computers are Internet-enabled, meaning they can access the Internet without wire connections. For example, a smart phone is a cell phone that connects to the Internet to allow users to transmit and receive e-mail messages, send text messages and pictures, and browse through Web sites on the phone display screen.

Workstations:

A workstation is a high-performance single-user computer with advanced input, output, and storage components that can be networked with other workstations and larger computers. Workstations are typically used for complex applications that require considerable computing power and high-quality graphics resolution, such as computer-aided design (CAD), computer-assisted manufacturing (CAM), desktop publishing, and software development. **Midrange Servers:**

Linked computers and terminals are typically connected to a larger and more powerful

computer called a network server, sometimes referred to as a host computer. Although the size and capacity of network servers vary considerably, most are midrange rather than large mainframe computers.

(i) Midrange server – formerly known as a minicomputer, a midrange server is a powerful computer capable of accommodating hundreds of client computers or terminals (users) at the same time.

(ii) Terminal – a device consisting of only a monitor and keyboard, with no processing capability of its own.

Mainframe Computers:

Larger, more powerful, and more expensive than midrange servers, a mainframe computer is capable of accommodating hundreds of network users performing different computing tasks. These computers are useful for dealing with large, ever-changing collections of data that can be accessed by many users simultaneously. Government agencies, banks, universities, and insurance companies use mainframes to handle millions of transactions each day.

- **Supercomputers:**

A supercomputer is the fastest, most powerful, and most expensive of all computers. Many

are capable of performing trillions of calculations in a single second. Primary applications include weather forecasting, comparing DNA sequences, creating artificially intelligent robots, and performing financial analyses.

---

7. Describe briefly about Secondary storage devices? (MAY 2009)

Secondary storage devices, as indicated by the name, save data after it has been saved by the primary storage device, usually referred to as RAM (Random Access Memory). From the moment we start typing a letter in Microsoft Word, for example, and until we click on "Save," your entire work is stored in RAM. However, once you power off your machine, that work is completely erased, and the only copy remaining is on the secondary storage device where we saved it, such as internal or external hard disk drive, optical drives for CDs or DVDs, or USB flash drive.

Internal Hard Disk Drive

The internal hard disk drive is the main secondary storage device that stores all of your data

magnetically, including operating system files and folders, documents, music and video. The hard disk drive is a stack of disks mounted one on top of the other and placed in a sturdy case. They are spinning at high speeds to provide easy and fast access to stored data anywhere on a disk.

External Hard Disk Drive

External hard disk drives are used when the internal drive does not have any free space and

you need to store more data. In addition, it is recommended to always back up all of our data and an external hard drive can be very useful, as they can safely store large amounts of information. They can be connected by either USB connection to a computer and can even be connected with each other in case you need several additional hard drives at the same time.

Optical Drive

An optical drive uses lasers to store and read data on CDs and DVDs. It basically burns a series of bumps and dips on a disc, which are associated with ones and zeros. Then, this same drive can interpret the series of ones and zeros into data that can be displayed on your monitors. There are a few different types of both CD and DVD disks, but the main two types include R and RW, which stand for Recordable (but you can write information on it just once) and Rewritable (meaning you can record data on it over and over again).

USB Flash Drive

USB flash memory storage device is also portable and can be carried around on a key chain. This type of a secondary storage device has become incredibly popular due to the very small size of device compared to the amount of data it can store (in most cases, more than CDs or DVDs). Data can be easily read using the USB (Universal Serial Bus) interface that now comes standard with most of the computers.

---

8. Explain about memory in Computer System?

(or)

Write short notes on memory of a computer? (MAY

2009)

The Role of Memory

The term "memory" applies to any electronic component capable of temporarily storing data. There are two main categories of memories:

Internal memory that temporarily memorizes data while programs are running. Internal memory uses micro conductors, i.e. fast specialized electronic circuits. Internal memory corresponds to what we call random access memory (RAM).

Auxiliary memory (also called physical memory or external memory) that stores information over the long term, including after the computer is turned off. Auxiliary memory corresponds to magnetic storage devices such as the hard drive, optical storage devices such as CD-ROMs and DVD-ROMs, as well as read-only memories.

Technical Characteristics

(a) Capacity, representing the global volume of information (in bits) that the memory can store

(b) Access time, corresponding to the time interval between the read/write request and the availability of the data

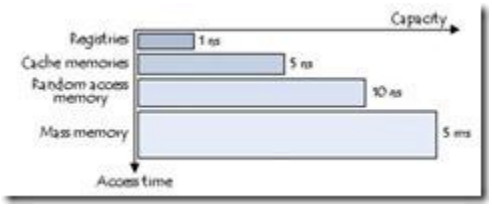
(c) Cycle time, representing the minimum time interval between two successive accesses

(d) Throughput, which defines the volume of information exchanged per unit of time, expressed in bits per second

(e) **Non-volatility,** which characterizes the ability of a memory to store data when it is not being supplied with electricity

The ideal memory has a large capacity with restricted access time and cycle time, a high throughput and is non-volatile.

However, fast memories are also the most expensive. This is why memories that use different technologies are used in a computer, interfaced with each other and organised hierarchically.



The fastest memories are located in small numbers close to the processor. Auxiliary memories, which are not as fast, are used to store information permanently.

Types of Memories

Random Access Memory

Random access memory, generally called RAM is the system's main memory, i.e. it is a space that allows you to temporarily store data when a program is running.

Unlike data storage on an auxiliary memory such as a hard drive, RAM is volatile, meaning that it only stores data as long as it supplied with electricity. Thus, each time the computer is turned off, all the data in the memory are irremediably erased.

Read-Only Memory

Read-only memory, called ROM, is a type of memory that allows you to keep the information contained on it even when the memory is no longer receiving electricity. Basically, this type of memory only has read-only access. However, it is possible to save information in some types of ROM memory.

Flash Memory

Flash memory is a compromise between RAM-type memories and ROM memories. Flash memory possesses the non-volatility of ROM memories while providing both read and writes access However, the access times of flash memories are longer than the access times of RAM.

---

9. Elaborate the various Input and Output Devices?

Input/Output devices are required for users to communicate with the computer. In simple terms, input devices bring information INTO the computer and output devices bring information OUT of a computer system. These input/output devices are also known as peripherals since they surround the CPU and memory of a computer system.

Some commonly used Input/Output devices are listed in table below.

Input Devices	Output Devices
Keyboard	Monitor
Mouse	LCD
Joystick	Printer
Scanner	Plotter
Light Pen	
Touch Screen	

(a) Keyboard

It is a text base input device that allows the user to input alphabets, numbers and

other characters. It consists of a set of keys mounted on a board.

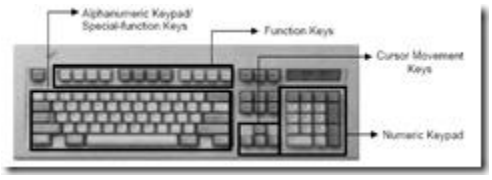


Figure 1: The Keyboard

Alphanumeric Keypad

It consists of keys for English alphabets, 0 to 9 numbers, and special characters like + − / * ( ) etc.

Function Keys

There are twelve function keys labeled F1, F2, F3… F12. The functions assigned to these keys differ from one software package to another. These keys are also user programmable keys.

Special-function Keys

These keys have special functions assigned to them and can be used only for those specific purposes. Functions of some of the important keys are defined below.

Enter

It is similar to the 'return' key of the typewriter and is used to execute a command or program.

Spacebar

It is used to enter a space at the current cursor location.

Backspace

This key is used to move the cursor one position to the left and also delete the character in that position.

Delete

It is used to delete the character at the cursor position.

Insert

Insert key is used to toggle between insert and overwrite mode during data entry.

Shift

This key is used to type capital letters when pressed along with an alphabet key. Also used to type the special characters located on the upper-side of a key that has two characters defined on the same key.

Caps Lock

Cap Lock is used to toggle between the capital lock features. When ‘on’, it locks the alphanumeric keypad for capital letters input only.

Tab

Tab is used to move the cursor to the next tab position defined in the document. Also, it is used to insert indentation into a document.

Ctrl

Control key is used in conjunction with other keys to provide additional functionality on the keyboard.

Alt

Also like the control key, Alt key is always used in combination with other keys to perform specific tasks.

Esc

This key is usually used to negate a command. Also used to cancel or abort executing programs.

Numeric Keypad

Numeric keypad is located on the right side of the keyboard and consists of keys having numbers (0 to 9) and mathematical operators (+ − * /) defined on them. This keypad is provided to support quick entry for numeric data.

Cursor Movement Keys

These are arrow keys and are used to move the cursor in the direction indicated by the arrow (up, down, left, right).

(b) Mouse

The mouse is a small device used to point to a particular place on the screen and select in order to perform one or more actions. It can be used to select menu commands, size windows, start programs etc. The most conventional kind of mouse has two buttons on top: the left one being used most frequently.

Mouse Actions

Left Click : Used to select an item.

Double Click : Used to start a program or open a file.

Right Click : Usually used to display a set of commands.

Drag and Drop : It allows you to select and move an item from one location to another. To achieve this place the cursor over an item on the screen, click the left

mouse button and while holding the button down move the cursor to where you want to place the item, and then release it.

(c) Joystick

Figure 2: The Mouse

The joystick is a vertical stick which moves the graphic cursor in a direction the stick is moved. It typically has a button on top that is used to select the option pointed by the cursor. Joystick is used as an input device primarily used with video games, training simulators and controlling robots

Figure 3: The Joystick

(d)Scanner

Scanner is an input device used for direct data entry from the source document into the computer system. It converts the document image into digital form so that it can be fed into the computer. Capturing

information like this reduces the possibility of errors typically experienced during large data entry.

Figure 4: The Scanner

Hand-held scanners are commonly seen in big stores to scan codes and price information for each of the items. They are also termed the bar code readers.

(e) Bar codes

A bar code is a set of lines of different thicknesses that represent a number. Bar Code Readers are used to input data from bar codes. Most products in shops have bar codes on them. Bar code readers work by shining a beam of light on the lines that make up the bar code and detecting the amount of light that is reflected back

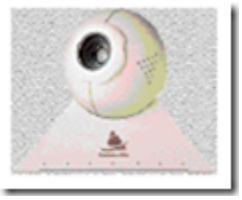
Figure 5: The Bar Code Reader

(f) Light Pen

It is a pen shaped device used to select objects on a display screen. It is quite like the mouse (in its functionality) but uses a light pen to move the pointer and select any object on the screen by pointing to the object. Users of Computer Aided Design (CAD) applications commonly use the light pens to directly draw on screen.

(g) Touch Screen

It allows the user to operate/make selections by simply touching the display screen. Common examples of touch screen include information kiosks, and bank ATMs. **(h)Digital camera**



A digital camera can store many more pictures than an ordinary camera. Pictures taken using a digital camera are stored inside its memory and can be transferred to a computer by connecting the camera to it. A digital camera takes pictures by converting the light passing through the lens at the front into a digital image.

Figure 6: The Digital camera

(i) The Speech Input Device

The "Microphones - Speech Recognition" is a speech Input device. To operate it we require using a microphone to talk to the computer. Also we need to add a sound card to the computer. The Sound card digitizes audio input into 0/1s .A speech recognition program can process the input and convert it into machine-recognized commands or input.

Output Devices

(a) Monitor

Monitor is an output device that resembles the television screen and uses a Cathode Ray Tube (CRT) to display information. The monitor is associated with a keyboard for manual input of characters and displays the information as it is keyed in. It also displays the program or application output. Like the television, monitors are also available in different sizes.

(b) Liquid Crystal Display (LCD)

LCD was introduced in the 1970s and is now applied to display terminals also. Its advantages like low energy consumption, smaller and lighter have paved its way for usage in portable computers (laptops).

(c) Printer

Figure 8: The LCD

Printers are used to produce paper (commonly known as hardcopy) output. Based on the technology used, they can be classified as Impact or Non-impact printers. Impact

printers use the typewriting printing mechanism wherein a hammer strikes the paper through a ribbon in order to produce output. Dot-matrix and Character printers fall under this category. Non-impact printers do not touch the paper while printing. They use chemical, heat or electrical signals to etch the symbols on paper. Inkjet, Deskjet, Laser, Thermal printers fall under this category of printers.

When we talk about printers we refer to two basic qualities associated with printers: resolution, and speed. Print resolution is measured in terms of number of dots per inch (dpi). Print speed is measured in terms of number of characters printed in a unit of time and is represented as characters-per-second (cps), lines-per-minute (lpm), or pages-per-minute (ppm).

Figure 9: The Printer

(d) Plotter

Plotters are used to print graphical output on paper. It interprets computer commands and makes line drawings on paper using multicolored automated pens. It is capable of producing graphs, drawings, charts, maps etc. Computer Aided Engineering (CAE) applications like CAD (Computer Aided Design) and CAM (Computer Aided Manufacturing) are typical usage areas for plotters.

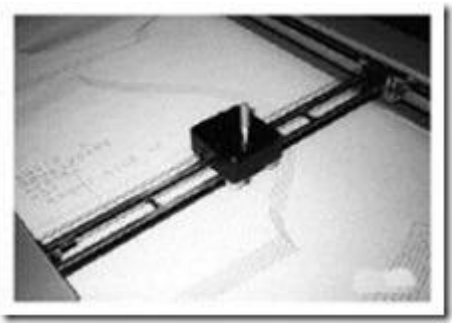


Figure 10: The Plotter

(e) Audio Output: Sound Cards and Speakers:

The Audio output is the ability of the computer to output sound. Two components

are needed: Sound card – Plays contents of digitized recordings, Speakers – Attached to sound card.

FUNDAMENTALS OF COMPUTING & COMPUTER PROGRAMMING

UNIT II

COMPUTER SOFTWARE

16 MARKS

---

1. Give the categories of Software with example? (JAN 2009/ MAY 2009)

SOFTWARE TYPES

Application Software:

Application Software is a set of programs for a specific application. Application software

is useful for word processing, accounting, and producing statistical report, Graphics, Excel and Data Base. Programming languages COBOL, FORTRAN, C++,VB, VC, Java

Types of Application Software

Application software enables users to perform the activities and work that computers were designed for. The specific type of application used depends on the intended purpose, and there are application programs for almost every need.

(a) Individual application software refers to programs individuals use at work or at home. Examples include word processing, spreadsheet, database management, and desktop publishing programs.

(b) Collaboration software (also called groupware) enables people at separate PC

workstations to work together on a single document or project, such as designing a new automobile engine.

(c) Vertical application software is a complete package of programs that work together

to perform core business functions for a large organization. For example, a bank might have a mainframe computer at its corporate headquarters connected to conventional terminals in

branch offices, where they are used by managers, tellers, loan officers, and other employees.

All financial transactions are fed to the central computer for processing. The system then generates managers' reports, account statements, and other essential documents.

Other Application Software Models

Shareware: Shareware is software developed by an individual or software publisher who retains ownership of the product and makes it available for a small "contribution" fee. The voluntary fee normally entitles users to receive online or written product documentation and technical help.

Freeware: Freeware is software that is provided free of charge to anyone wanting to use it. Hundreds of freeware programs are available, many written by college students and

professors who create programs as class projects or as part of their research.

Open Source Software: An open source software program is software whose programming code is owned by the original developer but made available free to the general public, who is

encouraged to experiment with the software, make improvements, and share the

improvements with the user community

Application Software for Individual Use

The thousands of application programs that individuals use to perform computing tasks at work and at home can be grouped into four types:

• Productivity software

• Software for household use

• Graphics and multimedia software

• Communication software

Productivity Software

Productivity software is designed to improve efficiency and performance on the job and at home, and is the largest category of application software for individual use.

Word Processing A word processing program can be used to create almost any kind of printed document. Word processors are the most widely used of all software applications because they

are central to communication. Whatever the type of document created with a word processing program, the essential parts of the procedure remain the same:

• create (enter) text

• edit the text

• format the document

• save and print the file

Desktop Publishing

Desktop publishing (DTP) software allows users to create impressive documents that include text, drawings, photographs, and various graphics elements in full color. Professional- quality publications can be produced with DTP software. Textbooks such as this one may be designed and laid out with a desktop publishing application such as PageMaker, QuarkXpress, or Adobe InDesign.

Spreadsheets

Spreadsheet software is an electronic version of the ruled worksheets accountants used in the past. Spreadsheet software provides a means of organizing, calculating, and presenting financial, statistical, and other numerical information. Businesses find spreadsheets particularly useful for evaluating alternative scenarios. By entering various data values and formulas into a spreadsheet, questions can be answered quickly and accurately.

For the individual user, spreadsheets fulfill many purposes, including:

• preparing and analyzing personal or business budgets

• reconciling checkbooks

• analyzing financial situations

• tracking and analyzing investments

• preparing personal financial statements

• estimating taxes

Database Management

In a computerized database system, data are stored in electronic form on a storage medium, such as hard or floppy disks or CDs. A database is a collection of data organized in one or more tables consisting of individual pieces of information, each located in a field, and a collection of related fields, each collection making up one record (see Figure 5-1). A commercial database program typically allows users to create a form for entering data. A user can design an electronic form to make entering information into the database easier. The information entered using such a form will become a record in a table. Users can add, remove, or change the stored data.

Presentation Graphics

Presentation graphics software allows users to create computerized slide shows that combine text, numbers, animation, graphics, sounds, and videos. A slide is an individual document that is created in presentation graphics software. A slide show may consist of any number of individual slides. For example, an instructor may use a slide show to accompany a lecture to make it more engaging and informative. Microsoft PowerPoint and Corel Presentations are two popular presentation software programs.

Software for Household Use

Numerous software applications designed for use in the household are available for purchase. Among the many products available are applications for managing personal finances, preparing tax returns, preparing legal documents, playing games, and education and reference.

Graphics and Multimedia Software

Graphics and multimedia software allows both professional and home users to work with graphics, video, and audio. A variety of applications software is focused in this area, including painting and drawing software, image-editing software, video and audio editing software, Web authoring software, and computer-aided design (CAD) software. **Communications Software**

One of the major reasons people use computers is to communicate with others and to retrieve and share information. Communications software allows users to send and receive e- mail, browse and search the Web, engage in group communications and discussions, and participate in videoconferencing activities.

1. Automatic Multimedia Tagging Software

2. Advances in Speech Recognition Software

3. Pattern Recognition Software

4. Distributed Computing

System Software:

(1) When you switch on the computer the programs written in ROM is executed which activates different units of your computer and makes it ready for you to work.

(2) This set of programs can be called system software.

(3) System software are general programs designed for performing tasks such as controlling all operations required to move data into and out of the computer

(4) System Software allows application packages to be run on the computer.

(5) Computer manufactures build and supply this system software with the computer system.

An operating system is the most important piece of software on a personal computer. The

location of the operating system identifies the boot drive for the personal computer, which is typically the hard drive. Once started, the operating system manages the computer system and performs functions related to the input, processing, output, and storage of information, including:

• Managing main memory, or RAM

• Configuring and controlling peripheral devices

• Managing essential file operations, including formatting or copying disks, and renaming or deleting files

• Monitoring system performance

• Providing a user interface

---

2. State different language translators and explain their functions?

(a) Compiler:

A **compiler** is a computer program (or set of programs) that transforms source code written in a programming language (the source language) into another computer language (the target language, often having a binary form known as object code).

(b) Loader:

In a computer operating system, a loader is a component that locates a given program (which can be an application or, in some cases, part of the operating system itself) in offline storage (such as a hard disk), loads it into main storage (in a personal computer, it's called random access memory), and gives that program control of the computer (allows it to execute its instruction s).

A program that is loaded may itself contain components that are not initially loaded into main storage, but can be loaded if and when their logic is needed. In a multitasking operating

system, a program that is sometimes called a dispatcher juggles the computer processor's time among different tasks and calls the loader when a program associated with a task is not already in main storage.

(c) Linker:

Also called link editor and binder, a linker is a program that combines object modules to form an executable program. Many programming languages allow you to write different pieces of code, called modules, separately. This simplifies the programming task because

you can break a large program into small, more manageable pieces. Eventually, though, you need to put all the modules together. This is the job of the linker. In addition to combining

modules, a linker also replaces symbolic addresses with real addresses. Therefore, you may need to link a program even if it contains only one module.

The linkage editor accepts two major types of input:

• Primary input, consisting of object decks and linkage editor control statements.

• Additional user-specified input, which can contain both object decks and control statements, or load modules. This input is either specified by you as input, or is incorporated automatically by the linkage editor from a call library.

Output of the linkage editor is of two types:

• A load module placed in a library (a partitioned data set) as a named member

• Diagnostic output produced as a sequential data set.

• The loader prepares the executable program in storage and passes control to it directly.

(d) Interpreter:

An **interpreter** normally means a computer program that executes, i.e. performs, instructions written in a programming language. An interpreter may be a program that either

Ø executes the source code directly

Ø translates source code into some efficient intermediate representation (code) and immediately executes this

Ø explicitly executes stored precompiled code made by a compiler which is part of the interpreter system

(e) Assembler:

An assembler translates an assembly language source program into machine codes. Though the assembly language is the symbolic representation of machine codes, a computer cannot understand it. After translating the assembly language program into machine codes by the assembler, the program becomes ready for the execution.

---

3. Explain in detail the steps involved in Software Development Process?

Software development life cycle model is also called as waterfall model which is followed by majority of systems. This software development life cycle process has the following seven stages in it namely

1. System Requirements Analysis

2. Feasibility study

3. Systems Analysis and Design

4. Code Generation

5. Testing

6. Maintenance

7. Implementation

1. System Requirements Analysis:

The first essential or vital thing required for any software development is system. Also the system requirement may vary based on the software product that is going to get developed. So a careful analysis has to be made about the system requirement needed for the development of the product. After the analysis and design of the system requirement phase the system required for the development would be complete and the concentration can be on the software development process.

2. Feasibility study:

After making an analysis in the system requirement the next step is to make analysis of the software requirement. In other words feasibility study is also called as software requirement analysis. In this phase development team has to make communication with customers and make analysis of their requirement and analyze the system. By making analysis this way it would be possible to make a report of identified area of problem. By making a detailed analysis on this area a detailed document or report is prepared in this phase which has details like project plan or schedule of the project, the cost estimated for developing and executing the system, target dates for each phase of delivery of system developed and so on. This phase is the base of software development process since further steps taken in software development life cycle would be based on the analysis made on this phase and so careful analysis has to be made in this phase.

3. Systems Analysis and Design:

This is an important phase in system development .Here analysis is made on the design of the system that is going to be developed. In other words database design, the design of the architecture chosen, functional specification design, low level design documents, high level design documents and so on takes place. Care must be taken to prepare these design documents because the next phases namely the development phase is based on these design documents. If a well structured and analyzed design document is prepared it would reduce the time taken in the coming steps namely development and testing phases of the software development life cycle.

4. **Code Generation:**

This is the phase where actual development of the system takes place. That is based on the design documents prepared in the earlier phase code is written in the programming technology chosen. After the code is developed generation of code also takes place in this phase. In other words the code is converted into executables in this phase after code generation.

5. **Testing:**

A software or system which is not tested would be of poor quality. This is because this is the phase where system developed would be tested and reports are

prepared about bugs or errors in system. To do this testing phase there are different

levels and methods of testing like unit testing, system test and so on. Based on the need the testing methods are chosen and reports are prepared about bugs. After this process the system again goes to development phase for correction of errors and again tested. This process continues until the system is found to be error free. To ease the testing process debuggers or testing tools are also available.

To develop reliable and good quality Program/Software we need to follow the following 5 steps :

1. Requirement Specification.

2. Analysis.

3. Design.

4. Implementation.

5. Verification and testing.

---

4. Write a short note on evolution of Internet? (FEB2009/FEB2010)

• The Internet is a network of networks

• Computer users on the Internet can contact one another anywhere in the world

• In Internet a huge resource of information is accessible to people across the world

• Information in every field starting from education, science, health, medicine, history, and geography to business, news, etc. can be retrieved through Internet

• You can also download programs and software packages from anywhere in the world

• In 1969 Department of Defense (DOD) of USA started a network called ARPANET (Advanced Research Projects Administration Network)

• Around 1970, NSFNET (National Science Foundation Network) was created. With the advancement of modern communication facilities,

• By 1990 many computers were looking up to NSFNET giving birth to Internet

• Internet is not a governmental organization.

• The ultimate authority of the Internet is the Internet Society.

• This is a voluntary membership organization whose purpose is to promote global information exchange.

• Internet has more than one million computers attached to it.

• Ten years of research brought Local Area Ethernet Networks (LANs) and workstations were developed to get connected to LAN.

• Computers connected to ARPANET used a standard or rule to communicate with each other with NCP (National Control Protocol).

• Protocol is a network term used to indicate the standard used by a network for communication.

• Rapid change in information technology suppressed NCP and brought TCP/IP (Transmission Control Protocol/Internet Protocol) in to the world of networking

• The Internet is a rare example of a large democracy with no state of head, no official censors, no bosses, and no board of directors. Nobody controls the Internet and in principle, any computer can speak to any other computer, as long as it obeys the technical rules of the TCP/IP protocol.

• This freedom of Internet helped it to move out of its original base in military and research institutions, into elementary and high schools, colleges, public libraries, commercial sectors.

5. Explain various types of Internet Connections?

(a) Cable Modem Broadband

A connection through an ordinary coax cable through your digital cable provider is the

easiest and most common way to connect to the Internet at high speeds. Most connections average about 400K/second download and 128K upload. Cable's largest advantage is its availability and ability to produce multiple upstreams (when sending). The biggest downside to a cable connection is the slow-downs you'll experience during gluts of service when several people are sharing the network. Expect to pay around $40 per month for this service.

(b) Digital Subscriber Link (DSL)

This is a connection using your phone line and a special modem. You have to be within so

many feet of a phone station "hub" and your line has to be of a newer type to qualify. Good portions of the population (especially in urban areas) match these criteria and can get a DSL connection. The modem uses a sound frequency well above the human ear's limits and will not interfere with normal telephone operation. Most connections average about 400-650K per second in download (some are faster) while anywhere from 128-256K in upload speed is available as well. The biggest downside to this type of connection is the availability. The biggest boon to this technology is its reliability and that network slow-downs are less common than with a cable connection.

(c) Satellite (HST)

This is the most expensive alternative for getting a high-speed connection to the Internet.

These come in two varieties, 1-way and 2-way. One-way satellites are like television receivers: they only accept signals. You'll still have to use your modem to connect for uploads. A 2-way connection, however, both sends and receives and is telephone-free. Average speeds for this type of connection are 600K and higher for download and 128K for the upload. Averages tend to be higher because there are far fewer users on the network to slow things down. The biggest up side to this technology is that it is available just about everywhere. The biggest downside to this type of connection is two-fold: price and reliability. Expect to spend $600 or more for the equipment and another $50 or more a month for the connection.

---

6. Define various Internet Terminologies? (At least 15 terms) (MAY 2009)

1. Modem

2. Web page

3. Web browser

4. Web site

5. Blogs

6. Home page

7. URL

8. IP Address

9. ISP

10. WWW

11. Intranet

12. Internet protocol

13. Domain Name System

14. Web sever

15. Email

16. Email Address

17. Hyperlink

18. Usenet

19. Internet Search Engine

20. Internet Chat

---

7. Explain various Internet applications?

(or)

Explain the Internet services in detail. (MAY 2009 / FEB2010)

1. World Wide Web:

World Wide Web is abbreviated as WWW, web or W3. This is a multimedia service which is most popular on internet. WWW content displays as a page. Along with text formatted in various fonts, styles, colors, and sizes, the pages may also contain pictures, images, animation, sound, video, movies in a single interface. The WWW is the fast growing part of the internet.

2. Rich Internet Application

Rich Internet Applications (RIA) are web applications that have the features and functionality of traditional desktop applications. RIA's typically transfer the processing necessary for the user interface to the web client but keep the bulk of the data (i.e maintaining the state of the program, the data etc) back on the application server.

RIA's typically:

• run in a web browser, or do not require software installation

• run locally in a secure environment called a sandbox

3. Electronic Mail (E-mail)

E-mail is the fast, easy and inexpensive way to communicate with other internet users around the world. E-mail can also be used to send or receive documents, images, audio-video etc., as an attachment along with the mail.

4. File Transfer Protocol:

FTP is a method of transferring files from one computer to another, connected on the internet. It is system of rules and software program that enables user to transfer information between computers. The uploading and downloading of files from the remote computer is possible using FTP if the remote machine access is permitted.

5. Telnet:

The word ‘telnet’ is derived by combining the words telecommunication and network. Telnet is a protocol which provides the capability to log onto a remote computer. Hence it is called a

‘remote login’. When you connect your computer to another computer using telnet, we can interact with another computer. The computer on which we are working is called as a local

computer.

6. Internet Relay Chat: (IRC):

IRC is the service provided by the internet to allow users on the internet to communicate and carry on conversations with other users via the computer. The simultaneous online conversations with other users from anywhere in the world are possible using IRC. **Requirements for IRC:**

1. The users for communication must be connected to the internet at the same time.

2. They must run the right software.

3. They must actively participate in chatting

Types of conversations:

Public

Allows every user in the channel to see what the user types.

Private

Allows to see messages only to two users who may or may not be on same channel.

Examples: mIRC, Virc, LeafChat.

7. Chatting and Instant Messaging:

The users on the internet communicate with each other by tying in real time. This is called chatting. The chat programs allow chatting.

Advantages of chatting:

• It is quite cheap.

• Unlike E-mail, it is done on real time.

• Unlike IRC, the user does not need to have any special software to connect to any caht rooms.

8. Internet Telephony:

The internet telephony is used to exchange telephonic information using internet. It needs hardware and software. When the internet is used as a transmission medium with requested hardware and software for telephone calls, it is called 'internet telephony'.

It consists of:

• **End Devices:** Traditional telephones or audio equipped personal computers.

• **Gatekeepers:** provide call admission control, bandwidth management; address translation, authentication and user location.

• **Voice Over Internet Protocol (VOIP):** This is required for communication.

9. Video Conferencing:

The video conferencing system allows the users at remote locations to communicate using a combination of text, audio and video information. Thus, it enables direct face-to-face communication across networks.

Types:

• **Point-to-point:** It allows communication between two parties at remote locations.

• **Multi-point:** It allows communication which involves more than two parties.

Communication Tools:

• Camera

• Visual Display

• Audio System

• Microphones

• Loud Speakers

• Compression

• User Interface

• Control System

10. Commerce through Internet:

Electronic communication technologies are used to transmit business information and transact business. This type of business model is called electronic Commerce or E-commerce or EC. Buying and selling of goods and services online is called E-commerce.

The information is exchanged digitally to conduct the business which includes Electronic

Data Interchange (EDI) AND Electronic Funds Transfer (EFT).

Advantages of E-commerce:

• It has established an electronic global worldwide market.

• The market operates 24 hours and 365 days a year.

• Organizations and individuals are able to market goods and services.

• The purchasing is supported by secure facilities such as digital signature and encryption.

11. Newsgroups: (Usenet):

Newsgroups are international groups which concentrate on a particular topic and try to gather information about the topic. Newsgroups usually describe topical discussion groups and not the traditional 'News'. Thus newsgroups provide a source of information. For any particular topic, the interested people write news on that topic and post the written information or articles to the newsgroups. The other interested people can read, reply and comment on them.

Thus, newsgroups act as a medium through which users can get information, they can take part in the discussion on the interested topic and can ask questions from the internet community.

12. Mailing Lists (List server):

The mailing list is a method of internet communication which enables people with similar interests from all over the world to communicate and share information with each other. The administration of the mailing list groups is performed by software called 'list server'.

A list server continuously observes for incoming mail on a certain mailbox. If any message is received, the listener forwards it to a list of other addresses. The user must have an

E-mail account and list server software loaded on his computer. The user has to send a

message from his E-mail account to the desired list server located on computer networks throughout the world. When the user subscribes to a list server, messages from the other

subscribers are automatically sent to his/her E-mail address.

---

8. Brief the major problems encountered in the software system?

(a) Correctness:

The correctness of the software system refers to

• Agreement of program code with specification.

• Independence of the actual application of the software system.

(b) Reliability

Reliability of a software system is defined as the probability that this system fulfills a

function for a specified number of input trails under specified input conditions in a specified time interval.

Also if the test produces the lower error rate the system is reliable. The error rate depends on the frequency of inputs and on the probability that an individual input will lead to error.

(c) User Friendliness

Adequacy

• The input required for the user should be limited to only what is necessary.

• The performance offered by the software system should be adapted to extensibility.

• The results that a software system delivers should be output in a clear and well- structured form and be easy be interpret.

Learnability

• The user interface should present more information as close to reality as possible and permit efficient utilization of the software's failures.

• The user manual should be structured clearly and simply.

(d) Robustness

A software system is robust if the consequences of an error in its operation, in the input, or in the hardware, in relation to a given application, and inversely proportional to the probability of the occurrence of this error in the given application.

(e) Maintainability

The maintainability of the software depends on

• Readability

• Extensibility

• Testability

(f) Readability

It depends on

• Form of representation

• Programming style

• Consistency

• Structure of the program

• Programming language used

• Quality of documentation

• Tools available for inspection

(e) Extensibility

It depends on

• Structure of the software system

• Readability of the code

• Availability of program documentation

• Implementation program

(f) Testability

Allows to debugging of the program during execution. It depends on

Modularity: well structured programs suitable for stepwise testing

Structuredness: Useful for systematic testing of all components.

(g) Efficiency

Ability of a software system to fulfill its purpose with the best possible

utilization of all necessary resources (time, storage, transmission, channels and peripherals)

(h) Portability

A software system which can be adapted to run on computers other than the one which it was designed.

It depends on

i. Degree of hardware independence ii. Implementation language

iii. Specialized system functions

iv. Hardware properties

v. System dependent elements are collected in easily interchangeable program components.

FUNDAMENTALS OF COMPUTING & COMPUTER PROGRAMMING

UNIT III

16 Marks

PROBLEM SOLVING AND OFFICE AUTOMATION

---

1. Explain the steps involved in developing a program with neat diagram? (FEB2009)

(or)

Brief about planning the computer program?

The Programming Process – Purpose

1. Understand the problem

v Read the problem statement

v Question users

v Inputs required

v Outputs required

v Special formulas

v Talk to users

2. Plan the logic

(a) ***Visual Design Tools***

v Input record chart

v Printer spacing chart

v Hierarchy chart

v Flowchart

(b) ***Verbal Design Tools***

v Narrative Description

v Pseudocode

3. Code the program

v Select an appropriate programming language

v Convert flowchart and/or Pseudocode instructions into programming language statements

4. Test the program

1. Syntax errors

2. Runtime errors

3. Logic errors

4. Test Data Set

5. Implement the program

Buy hardware Publish software Train users

6. Maintain the program

Maintenance programmers

Legacy systems

Up to 85% of IT department budget

---

2. Explain flowchart in detail? (FEB 2009/FEB 2010) Definitions:

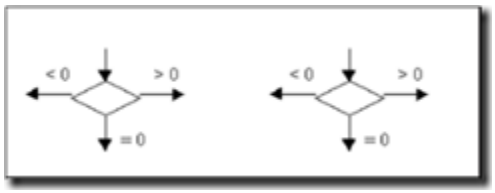
A flowchart is a schematic representation of an algorithm or a stepwise process, showing the steps as boxes of various kinds, and their order by connecting these with

arrows. Flowcharts are used in designing or documenting a process or program.

A flow chart, or flow diagram, is a graphical representation of a process or system that details the sequencing of steps required to create output. A flowchart is a picture of the separate steps of a process in sequential order.

The benefits of flowcharts are as follows:

a. **Communication:** Flowcharts are better way of communicating the logic of a system to all concerned.



b. **Effective analysis:** With the help of flowchart, problem can be analyzed in more effective way.

c. **Proper documentation:** Program flowcharts serve as a good program documentation, which is needed for various purposes.

d. **Efficient Coding:** The flowcharts act as a guide or blueprint during the systems analysis and program development phase.

e. **Proper Debugging:** The flowchart helps in debugging process.

f. **Efficient Program Maintenance:** The maintenance of operating program becomes easy with the help of flowchart. It helps the programmer to put efforts more efficiently on that part.

Advantages

Logic Flowcharts are easy to understand. They provide a graphical representation of actions to be taken.

Logic Flowcharts are well suited for representing logic where there is intermingling among many actions.

Disadvantages

Logic Flowcharts may encourage the use of GoTo statements leading to software design that is unstructured with logic that is difficult to decipher.

Without an automated tool, it is time-consuming to maintain Logic Flowcharts. Logic Flowcharts may be used during detailed logic design to specify a module.

However, the presence of decision boxes may encourage the use of GoTo statements, resulting in software that is not structured. For this reason, Logic Flowcharts may be better used during Structural Design.

LIMITATIONS OF USING FLOWCHARTS

Complex logic: Sometimes, the program logic is quite complicated. In that case, flowchart becomes complex and clumsy.

Alterations and Modifications: If alterations are required the flowchart may require re-

drawing completely.

Reproduction: As the flowchart symbols cannot be typed, reproduction of flowchart becomes a problem.

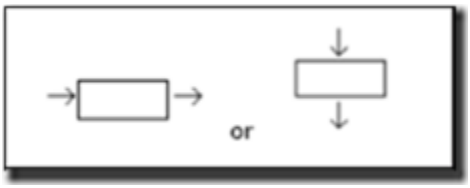
The essentials of what is done can easily be lost in the technical details of how it is done.

GUIDELINES FOR DRAWING A FLOWCHART

a. In drawing a proper flowchart, all necessary requirements should be listed out in logical order.

b. The flowchart should be clear, neat and easy to follow. There should not be any room for ambiguity in understanding the flowchart.

c. The usual direction of the flow of a procedure or system is from left to right or top to bottom.



d. Only one flow line should come out from a process symbol.

e. Only one flow line should enter a decision symbol, but two or three flow lines, one for each possible answer, should leave the decision symbol.

f. Only one flow line is used in conjunction with terminal symbol.

g. Write within standard symbols briefly. As necessary, you can use the annotation symbol to describe data or computational steps more clearly.

h. If the flowchart becomes complex, it is better to use connector symbols to reduce the number of flow lines. Avoid the intersection of flow lines if you want to make it more effective and better way of communication.

i. Ensure that the flowchart has a logical *start* and *finish.*

j. It is useful to test the validity of the flowchart by passing through it with a simple test data.

---

3. Describe in detail about algorithm? Give example (MAY 2009)

An algorithm is a description of a procedure which terminates with a result. Simple algorithms can be implemented within a function.

Properties of an algorithm

No ambiguity

There should not be any uncertainty about which instruction to execute next. The algorithm should conclude after a finite number of steps.

The algorithm should be general.

Example:

Biggest among two numbers Addition of N numbers Finding Factorial of a number Finding Fibonacci series.

---

4. Elaborate pseudo code with example. (MAY 2009\FEB 2009) Guidelines

Statements should be in simple English

Must produce a solution for the specified problem

It should be concise

Each instruction should be written in separate line and express one action. Capitalize keywords such as READ, PRINT and so on.

Instruction should be written from top to bottom, with one entry and one exit. Should allow easy transition from design to coding in programming language.

Benefits:

Language Independent.

Easy to develop a program from pseudo code than flowchart. Easy to translate into programming language.

It is compact.

Limitations:

No visual representation of program logic

No accepted standards for writing pseudo codes. Cannot be compiled or executed.

No real form or syntax rules.

Examples:

Finding a number is prime or not

---

5. Discuss about the program control structure and program paradigms in detail.

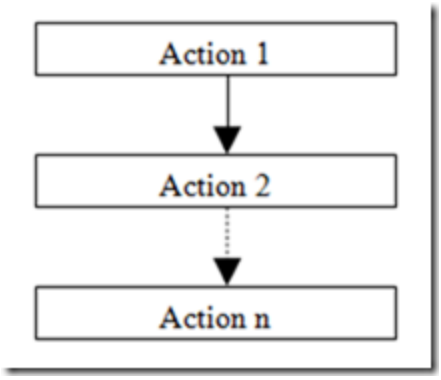
Program structures that affect the order in which statements are executed or that affect statements are executed are called control structures.

1. **Sequence control structure**

Instructions has to follow one another in a logical progression is sequence control structure.

Example:

Addition of two numbers

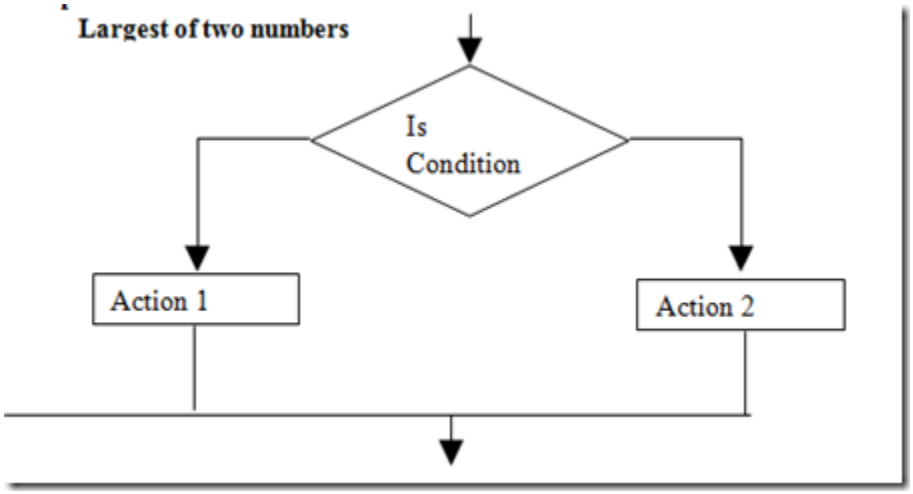


2. Selection Control Structure

Selection control structure allows the program to make a chance between alternate paths, when it is true or false.

Example:

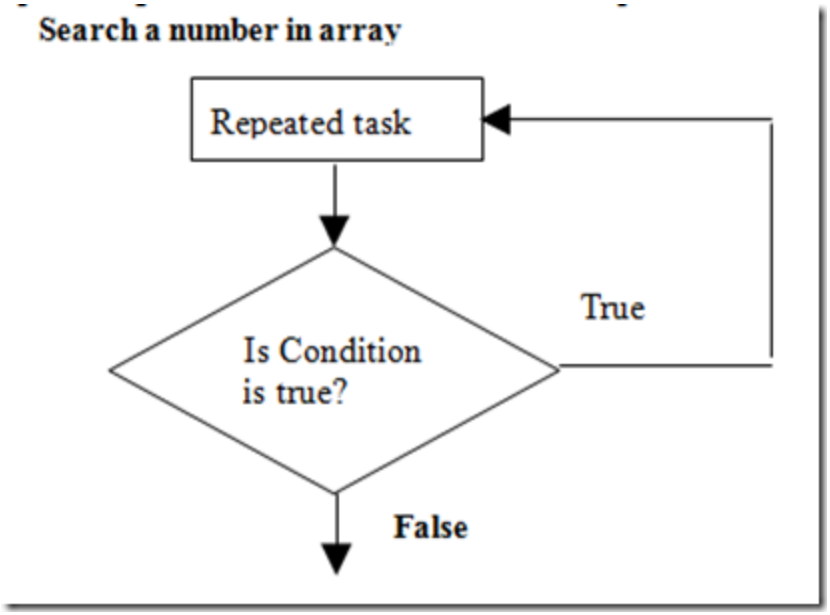
Largest of two numbers



3. Repetition Control Structure

Directs the system to loop back to a previous statement in the program, repeating the same sequence over and again, usually with a new data. When a sequence of statements is repeated against a condition, it is said to a loop.

Example: Search a number in array



---

6. Explain in detail about the word processing package. (OR)

Explain 8 formatting features in word processing package. (JAN 2009/MAY 2009)

(1) Paragraph

(2) Font

(3) Bullets and Numbering (4) Boarders and Shading (5) Tabs

(6) Change case

(7) Auto format

(8) Background

---

7. Describe about spreadsheet packages. (OR)

Explain the formatting features in spreadsheet package. (JAN 2009/MAY 2009)

(1) Cells

(2) Rows

(3) Columns

(4) Auto format

(5) Sheet

(6) Conditional formatting

(7) Style

---

8. Discuss about the graphics package along with its various features in detail. (OR)

Explain 8 formatting features in graphics package. (JAN 2009/MAY 2009)

(1) Fonts

(2) Bullets and Numbering

(3) Alignment

(4) Line spacing (5) Change case (6) Replace fonts

(7) Slide Design

(8) Slide Layout

FUNDAMENTALS OF COMPUTING & COMPUTER PROGRAMMING

UNIT IV – 16 Marks

INTRODUCTION TO C

---

1. Explain in detail about 'C' declarations and variables.

In C, lowercase and uppercase characters are very important. All commands in C

must be lowercase. The C programs starting point is identified by the word main( ). This informs the computer as to where the program actually starts.

The brackets that follow the keyword main indicate that there are no arguments supplied to this program.

The two braces, { and }, signify the begin and end segments of the program. The purpose of the statement

include <stdio.h> is to allow the use of the printf statement to provide program

output. Text to be displayed by printf() must be enclosed in double quotes. The program has only one statement printf("Programming in C is easy.\n");

printf() is actually a function (procedure) in C that is used for printing variables and text. Where text appears in double quotes "", it is printed without modification. There are some exceptions however. This has to do with the \ and % characters. These characters are modifier's, and for the present the \ followed by the n character represents a newline character. Thus the program prints

Programming in C is easy.

and the cursor is set to the beginning of the next line. As we shall see later on, what follows the \ character will determine what is printed, ie, a tab, clear screen, clear line etc. Another important thing to remember is that all C statements are terminated by a semi-colon ;

General rules of 'C' language:

· program execution begins at main()

· keywords are written in lower-case

· statements are terminated with a semi-colon

· text strings are enclosed in double quotes

· C is case sensitive, use lower-case and try not to capitalize variable names

· \n means position the cursor on the beginning of the next line

· printf() can be used to display text to the screen

· The curly braces {} define the beginning and end of a program block.

BASIC STRUCTURE OF C PROGRAMS

C programs are essentially constructed in the following manner, as a number of well defined sections.

```
/* HEADER SECTION */
/* Contains name, author, revision number*/
/* INCLUDE SECTION */
/* contains #include statements */
/* CONSTANTS AND TYPES SECTION */
/* contains types and #defines */
/* GLOBAL VARIABLES SECTION */
/* any global variables declared here */
/* FUNCTIONS SECTION */
/* user defined functions */
/* main() SECTION */
int main()
{
}
```

A Simple Program

The following program is written in the C programming language.

```
#include <stdio.h>

main()

{

printf("Programming in C is easy.\n"); }
```

INITIALISING DATA VARIABLES AT DECLARATION TIME

In C, variables may be initialized with a value when they are declared. Consider

the following declaration, which declares an integer variable count which is initialized to

10. int count = 10;

SIMPLE ASSIGNMENT OF VALUES TO VARIABLES

The **=** operator is used to assign values to data variables. Consider the following statement, which assigns the value 32 an integer variable count, and the letter **A** to the character variable letter

```
count = 32;

letter = 'A'
```

Variable Formatters

%d decimal integer

%c character

%s string or character array

%f float

%e double

HEADER FILES

Header files contain definitions of functions and variables which can be incorporated into any C program by using the pre-processor #include statement. Standard header files are provided with each compiler, and cover a range of areas, string handling, mathematical, data conversion, printing and reading of variables.

To use any of the standard functions, the appropriate header file should be

included. This is done at the beginning of the C source file. For example, to use the function printf() in a program, the line

#include <stdio.h> should be at the beginning of the source file, because the definition for printf() is found in the file stdio.h All header files have the extension .h and generally reside in the /include subdirectory.

#include <stdio.h>

#include “mydecls.h”

The use of angle brackets <> informs the compiler to search the compilers include directory for the specified file. The use of the double quotes “” around the filename inform the compiler to search in the current directory for the specified file.

---

2. Explain in detail about the constants, expressions and statements in ‘C’.

1. Constants: (with examples)

1**.** Numeric constants

a. Integer Constants

b. Real Constants

2. Character constants

a. Single character Constants b. String Constants

2. Expressions:

An expression represents a single data item, such as number or a character. Logical conditions that are true or false are represented by expressions.

Example: a = p – q / 3 + r * 2 - 1

3. Statements

· Assignment Statements – Definition and examples

· Null Statements – Definition and examples

· Block of statements – Definition and examples

· Expression statements – Definition and examples

· Declaration statements – Definition and examples

---

3. Discuss about the various data types in 'C'. (MAY 2009)

The four basic data types are

a. INTEGER

These are whole numbers, both positive and negative. Unsigned integers

(positive values only) are supported. In addition, there are short and long integers.

The keyword used to define integers is, int

An example of an integer value is 32. An example of declaring an integer variable called **sum** is,

int sum;

sum = 20;

b. FLOATING POINT

These are numbers which contain fractional parts, both positive and negative. The keyword used to define float variables is,

float

An example of a float value is 34.12. An example of declaring a float variable

called **money** is,

float money;

money = 0.12;

c. DOUBLE

These are exponentional numbers, both positive and negative. The keyword used

to define double variables is, double

An example of a double value is 3.0E2. An example of declaring a double variable called **big** is,

double big;

big = 312E+7;

d. CHARACTER

These are single characters. The keyword used to define character variables is,

char

An example of a character value is the letter **A**. An example of declaring a character variable called **letter** is,

char letter;

letter = 'A';

Note the assignment of the character A to the variable letter is done by enclosing the value in **single quotes**.

#include < stdio.h >

main()

{

int sum;

float money; char letter; double pi;

sum = 10; /* assign integer value */ money = 2.21; /* assign float value */ letter = 'A'; /* assign character value */ pi = 2.01E6; /* assign a double value */ printf("value of sum = %d\n", sum);

printf("value of money = %f\n", money); printf("value of letter = %c\n", letter); printf("value of pi = %e\n", pi);

}

Sample program output

value of sum = 10

value of money = 2.210000 value of letter = A

value of pi = 2.010000e+06

4. Describe the various types of operators in 'C' language along with its priority.

An ex pr e s s I o n is a sequence of operators and operands that specifies computation of a value, or that designates an object or a function, or that generates side effects, or that performs a combination thereof.

Operation	Operator	Comment	Value of Sum before	Value of sum after
Multiply	*	sum = sum * 2;	4	8
Divide	/	sum = sum / 2;	4	2
Addition	+	sum = sum + 2;	4	6
Subtraction	-	sum = sum -2;	4	2
Increment	++	++sum;	4	5
Decrement	--	--sum;	4	3
Modulus	%	sum = sum % 3;	4	1

1. ARITHMETIC OPERATORS:

The symbols of the arithmetic operators are:-

Example:

```
#include <stdio.h>

main()

{

int sum = 50; float modulus; modulus = sum % 10;

printf("The %% of %d by 10 is %f\n", sum, modulus);

}
```

PRE/POST INCREMENT/DECREMENT OPERATORS

PRE means do the operation first followed by any assignment operation. POST

means do the operation after any assignment operation. Consider the following statements ++count; /* PRE Increment, means add one to count */ count++; /* P OST Increment, means add one to count */

Example:

```
#include <stdio.h>

main()

{

int count = 0, loop;

loop = ++count; /* same as count = count + 1; loop = count; */

printf("loop = %d, count = %d\n", loop, count);

loop = count++; /* same as loop = count; count = count + 1; */

printf("loop = %d, count = %d\n", loop, count);

}
```

If the operator precedes (is on the left hand side) of the variable, the operation is performed first, so the statement

```
loop = ++count;
```

really means increment count first, then assign the new value of count to loop.

2. THE RELATIONAL OPERATORS

These allow the comparison of two or more variables.

= = equal to

! = not equal

< less than

< = less than or equal to

> greater than

> = greater than or equal to

Example:

```
#include <stdio.h>

main() /* Program introduces the for statement, counts to ten */
```

```
{

int count;

for( count = 1; count <= 10; count = count + 1 )

printf("%d ", count );

printf("\n");

}
```

3. LOGICAL OPERATORS (AND, NOT, OR, EOR) Combining more than one condition

These allow the testing of more than one condition as part of selection

statements. The symbols are

LOGICAL AND &&

Logical and requires all conditions to evaluate as TRUE (non-zero).

LOGICAL OR ||

Logical or will be executed if any ONE of the conditions is TRUE (non-zero).

LOGICAL NOT !

logical not negates (changes from TRUE to FALSE, vsvs) a condition.

LOGICAL EOR ^

Logical eor will be excuted if either condition is TRUE, but NOT if they are all true.

The following program uses an if statement with logical AND to validate the users input to be in the range 1-10.

```
#include <stdio.h>

main()

{

int number;

int valid = 0;
```

```
while( valid == 0 ) {
printf(“Enter a number between 1 and 10 à”);
scanf(“%d”, &number);
if( (number < 1 ) || (number > 10) ){
printf(“Number is outside range 1-10. Please re-enter\n”);
valid = 0;
}
else
valid = 1;
}
printf(“The number is %d\n”, number );
```

} **Example: NEGATION**

```
#include <stdio.h>
main()
{
int flag = 0;
if( ! flag ) {
printf(“The flag is not set.\n”);
flag = ! flag;
}
printf(“The value of flag is %d\n”, flag);
}
```

Example:

Consider where a value is to be inputted from the user, and checked for validity to be within a certain range, lets say between the integer values 1 and 100.

```
#include <stdio.h>

main()

{

int number;

int valid = 0;

while( valid == 0 ) {

printf("Enter a number between 1 and 100");

scanf("%d", &number );

if( (number < 1) || (number > 100) ) printf("Number is outside legal range\n"); else

valid = 1;

}

printf("Number is %d\n", number );

}
```

4. THE CONDITIONAL EXPRESSION OPERATOR or TERNARY OPERATOR

This conditional expression operator takes THREE operators. The two symbols used to denote this operator are the ? and the :. The first operand is placed before the ?, the second operand between the ? and the :, and the third after the :. The general format is,

condition ? expression1 : expression2.

If the result of condition is TRUE (non-zero), expression1 is evaluated and the

result of the evaluation becomes the result of the operation. If the condition is FALSE (zero), then expression2 is evaluated and its result becomes the result of the operation. An example will help,

s = (x < 0) ? -1 : x * x; If x is less than zero then s = -1

If x is greater than zero then s = x * x

Example:

```
#include <stdio.h>

main()

{

int input;

printf("I will tell you if the number is positive, negative or zero!"\n");

printf("please enter your number now-à");

scanf("%d", &input );

(input < 0) ? printf("negative\n") : ((input > 0) ? printf("positive\n") :

printf("zero\n"));

}
```

5. BIT OPERATIONS

Operation	Operator	Comment	Value of Sum before	Value of sum after
AND	&	sum = sum & 2;	4	0
OR	|	sum = sum | 2;	4	6
Exclusive OR	^	sum = sum ^ 2;	4	6
1's Complement	~	sum = ~sum;	4	-5
Left Shift	<<	sum = sum << 2;	4	16
Right Shift	>>	sum = sum >> 2;	4	0

C has the advantage of direct bit manipulation and the operations available are,

Example:

```
/* Example program illustrating << and >> */

#include <stdio.h>

main()

{
```

```
int n1 = 10, n2 = 20, I = 0;

I = n2 << 4; /* n2 shifted left four times */

printf("%d\n", i);

I = n1 >> 5; /* n1 shifted right five times */

printf("%d\n", i);

}
```

Example:

```
/* Example program using EOR operator */

#include <stdio.h>

main()

{

int value1 = 2, value2 = 4;

value1 ^= value2;

value2 ^= value1;

value1 ^= value2;

printf("Value1 = %d, Value2 = %d\n", value1, value2);

}
```

Example:

```
/* Example program using AND operator */

#include <stdio.h>

main()

{

int loop;
```

```
for( loop = 'A'; loop <= 'Z'; loop++ )

printf("Loop = %c, AND 0xdf = %c\n", loop, loop & 0xdf);

}
```

---

5. Explain about the various decision making statements in 'C' language.

(JAN 2009/FEB2010)

1. IF STATEMENTS

DECISION MAKING

The if statements allows branching (decision making) depending upon the value or state of variables. This allows statements to be executed or skipped, depending upon decisions.

The basic format is,

```
if( expression )

program statement;
```

Example:

```
if( students < 65 )

++student_count;
```

In the above example, the variable student_count is incremented by one only if the value of the integer variable students is less than 65. The following program uses an if statement to validate the users input to be in the range 1-10.

```
#include <stdio.h>

main()

{

int number;

int valid = 0;

while( valid == 0 ) {
```

```
printf(“Enter a number between 1 and 10 à”);

scanf(“%d”, &number);

/* assume number is valid */

valid = 1;

if( number < 1 ) {

printf(“Number is below 1. Please re-enter\n”);

valid = 0;

}

if( number > 10 ) {

printf(“Number is above 10. Please re-enter\n”);

valid = 0;

}

}

printf(“The number is %d\n”, number );

}
```

2. IF ELSE

The general format for these are,

```
if( condition 1 )

statement1;

else if( condition 2 )

statement2;

else if( condition 3 )

statement3; else statement4;
```

The else clause allows action to be taken where the condition evaluates as false (zero). The following program uses an if else statement to validate the users input to be in the range 1-10.

Example:

```
#include <stdio.h>

main()

{

int number;

int valid = 0;

while( valid == 0 ) {

printf("Enter a number between 1 and 10 à");

scanf("%d", &number);

if( number < 1 ) {

printf("Number is below 1. Please re-enter\n");

valid = 0;

}

else if( number > 10 ) {

printf("Number is above 10. Please re-enter\n");

valid = 0;

}

else

valid = 1;

}

printf("The number is %d\n", number );

}
```

This program is slightly different from the previous example in that an else clause is used to set the variable valid to 1. In this program, the logic should be easier to follow.

3. NESTED IF ELSE

/* Illustates nested if else and multiple arguments to the scanf function. */

Example:

```
#include <stdio.h>

main()

{

int invalid_operator = 0;

char operator;

float number1, number2, result;

printf("Enter two numbers and an operator in the format\n");

printf(" number1 operator number2\n");

scanf("%f %c %f", &number1, &operator, &number2);

if(operator == '*')

result = number1 * number2;

else if(operator == '/')

result = number1 / number2;

else if(operator == '+')

result = number1 + number2;

else if(operator == '-')

result = number1 – number2;

else

invalid_operator = 1;
```

```
if( invalid_operator != 1 )
printf("%f %c %f is %f\n", number1, operator, number2, result );
else
printf("Invalid operator.\n");
```

---

6. Write short notes on the following: (JAN 2009)

'for' loop

'while' loop

'dowhile' loop

'Switch case ' **(MAY 2009/FEB 2009/FEB 2010)**

BRANCHING AND LOOPING

1. ITERATION, FOR LOOPS

The basic format of the for statement is,

```
for( start condition; continue condition; re-evaulation )
program statement;
/* sample program using a for statement */
#include <stdio.h>
main() /* Program introduces the for statement, counts to ten */
{
int count;
for( count = 1; count <= 10; count = count + 1 )
printf("%d ", count );
printf("\n");
```

}

The program declares an integer variable count. The first part of the for statement **for (count = 1; initialized the value of count to 1.**

The for loop continues with the condition count <= 10; evaluates as TRUE. As

the variable count has just been initialized to 1, this condition is TRUE and so the

program statement printf("%d ", count); is executed, which prints the value of count to the screen, followed by a space character.

Next, the remaining statement of the for is executed count = count + 1); which adds one to the current value of count. Control now passes back to the conditional test, count <= 10; which evaluates as true, so the program statement printf("%d ", count); is executed.

Count is incremented again, the condition re-evaluated etc, until count reaches a value of

11.

When this occurs, the conditional test count <= 10; evaluates as FALSE, and the for loop terminates, and program control passes to the statement printf("\n"); which prints a newline, and then the program terminates, as there are no more statements left to execute.

2. THE WHILE STATEMENT

The while provides a mechanism for repeating C statements whilst a condition is

true. Its format is, while(condition) program statement;

Somewhere within the body of the while loop a statement must alter the value of the condition to allow the loop to finish.

Example:

```
/* Sample program including while */

#include <stdio.h>

main()

{

int loop = 0;

while( loop <= 10 ) {
```

```
printf("%d\n", loop);
++loop;
}
}
```

The above program uses a while loop to repeat the statements

printf("%d\n",loop); ++loop; the value of the variable loop is less than or

equal to 10.

3. THE DO WHILE STATEMENT

The do { } while statement allows a loop to continue whilst a condition evaluates as

TRUE (non-zero). The loop is executed as least once.

Example:

```
/* Demonstration of DO...WHILE */
#include <stdio.h>
main()
{
int value, r_digit;
printf("Enter the number to be reversed.\n");
scanf("%d", &value);
do {
r_digit = value % 10; printf("%d", r_digit); value = value / 10;
} while( value != 0 );
printf("\n");
}
```

The above program reverses a number that is entered by the user. It does this by using the modulus % operator to extract the right most digit into the variable r_digit. The original number is then divided by 10, and the operation repeated whilst the number is not equal to 0.

4. **SWITCH CASE:**

The switch case statement is a better way of writing a program when a series of if elses occurs.

The general format for this is, switch (expression) { case value1:

program statement;

program statement;

..... break;

case valuen:

program statement;

..... break; default:

.....

..... break;

}

The keyword break must be included at the end of each case statement. The default clause is optional, and is executed if the cases are not met. The right brace at the end signifies the end of the case selections.

Example:

#include <stdio.h>

main()

{

int menu, numb1, numb2, total;

printf("enter in two numbers à"); scanf("%d %d", &numb1, &numb2); printf("enter in choice\n"); printf("1=addition\n"); printf("2=subtraction\n");

scanf("%d", &menu);

```
switch( menu ) {

case 1: total = numb1 + numb2; break;

case 2: total = numb1 – numb2; break;

default: printf("Invalid option selected\n");

}

if( menu == 1 )

printf("%d plus %d is %d\n", numb1, numb2, total );

else if( menu == 2 )

printf("%d minus %d is %d\n", numb1, numb2, total );

}
```

The above program uses a switch statement to validate and select upon the users input choice, simulating a simple menu of choices.

---

7. Explain briefly about the input and output function in 'C'. (MAY 2009/FEB 2009)

1 printf ():

MANAGING INPUT AND OUTPUT OPERATORS

printf() is actually a function (procedure) in C that is used for printing variables and text. Where text appears in double quotes "", it is printed without modification. There are some exceptions however.

This has to do with the \ and % characters. These characters are modifiers, and

for the present the \ followed by the n character represents a newline character.

Example:

```
#include <stdio.h>

main()

{
```

printf(“Programming in C is easy.\n”);

printf(“And so is Pascal.\n”);

}

@ Programming in C is easy. And so is Pascal.

FORMATTERS for printf are, Cursor Control Formatters

\n newline

\t tab

\r carriage return

\f form feed

\v vertical tab

2. Scanf ():

Scanf () is a function in C which allows the programmer to accept input from a

keyboard.

Example:

#include <stdio.h>

main() /* program which introduces keyboard input */

{

int number;

printf(“Type in a number \n”);

scanf(“%d”, &number);

printf(“The number you typed was %d\n”, number);

}

FORMATTERS FOR scanf()

The following characters, after the % character, in a scanf argument, have the

following effect.

D read a decimal integer o read an octal value

x read a hexadecimal value h read a short integer

l read a long integer

f read a float value

e read a double value

c read a single character

s read a sequence of characters

[...] Read a character string. The characters inside the brackets

3. ACCEPTING SINGLE CHARACTERS FROM THE KEYBOARD Getchar, Putchar

getchar() gets a single character from the keyboard, and putchar() writes a single character from the keyboard.

Example:

The following program illustrates this,

```
#include <stdio.h>

main()

{

int i;

int ch;

for( i = 1; i<= 5; ++i ) { ch = getchar(); putchar(ch);

}

}
```

The program reads five characters (one for each iteration of the for loop) from the keyboard. Note that getchar() gets a single character from the keyboard, and putchar() writes a single character (in this case, ch) to the console screen.

---

8. (a) Describe in detail about type conversions in 'C' with example.

(b) Define delimiters. List them. Give an example program using various delimiters.

---

9. Explain the following:

· Keywords

· Identifiers

· C character set

· Constant and Volatile variables.

---

10. Explain the following:

· break statement with example program

· continue statement with example program

· goto statement with example program

FUNDAMENTALS OF COMPUTING & COMPUTER PROGRAMMING

UNIT V – 16 Marks

FUNCTIONS AND POINTERS

---

1. What are functions? Explain the types of functions in detail with an example program for each type.

A function is a self contained block or a sub program of one or more statements

that performs a special task when called.

Types:

· Library Functions

· User Defined functions

(a) Function Declaration

returntype function-name(Parameters);

Example:

int square(int, int);

(b) Function calling

function-name(actual parameters);

Example:

int square(a,b);

(c) Function Definition:

returntype function-name(formal parameters)

{

}

Example:

local variable declaration;

statement 1; statement 2; return(value);

void square(int a, int b)

{

printf("%d",(a*b));

}

Example for functions:

· Addition of two numbers where addition is a separate function

· Program using function for evaluating Fibonacci series.

---

2. Define arrays. Explain the array types with an example program for each type.

Arrays are data structures which hold multiple variables of the same data type. Consider the case where a programmer needs to keep track of a number of people within an organization. So far, our initial attempt will be to create a specific variable for each user.

This might look like,

int name1 = 101; int name2 = 232; int name3 = 231;

It becomes increasingly more difficult to keep track of this as the number of variables increase. Arrays offer a solution to this problem. An array is a multi-element box, a bit like a filing cabinet, and uses an indexing system to find each variable stored within it. In C, indexing starts at **zero**. Arrays, like other variables in C, must be declared before they can be used. The replacement of the above example using arrays looks like,

int names[4]; names[0] = 101; names[1] = 232; names[2] = 231; names[3] = 0;

We created an array called names, which has space for four integer variables. You may also see that we stored 0 in the last space of the array. This is a common technique used by C programmers to signify the end of an array. Arrays have the following syntax, using square brackets to access each indexed value (called an element).

x[i]

so that x[5] refers to the sixth element in an array called x. In C, array elements start with 0. Assigning values to array elements is done by,

x[10] = g; and assigning array elements to a variable is done by, g = x[10];

In the following example, a character based array named word is declared, and each element is assigned a character. The last element is filled with a zero value, to signify the end of the character string (in C, there is no string type, so character based arrays are used to hold strings). A printf statement is then used to print out all elements of the array.

```
/* Introducing array's, 2 */

#include <stdio.h>

main()

{

char word[20]; word[0] = 'H'; word[1] = 'e'; word[2] = 'l'; word[3] = 'l'; word[4] = 'o'; word[5] = 0;

printf("The contents of word[] is -->%s\n", word );

}
```

DECLARING ARRAYS

Arrays may consist of any of the valid data types. Arrays are declared along with all other variables in the declaration section of the program.

```
/* Introducing array's */

#include <stdio.h>

main()

{

int numbers[100]; float averages[20]; numbers[2] = 10;

--numbers[2];

printf("The 3rd element of array numbers is %d\n", numbers[2]);

}
```

The above program declares two arrays, assigns 10 to the value of the 3rd element of array numbers, decrements this value (--numbers[2]), and finally prints the value. The number of elements that each array is to have is included inside the square brackets

ASSIGNING INITIAL VALUES TO ARRAYS

The declaration is preceded by the word static. The initial values are enclosed in braces,

Example:

```
#include <stdio.h>

main()

{

int x;

static int values[] = { 1,2,3,4,5,6,7,8,9 };

static char word[] = { 'H','e','l','l','o' };

for( x = 0; x < 9; ++x )

printf("Values [%d] is %d\n", x, values[x]);

}
```

MULTI DIMENSIONED ARRAYS

Multi-dimensioned arrays have two or more index values which specify the element in the array.

multi[i][j];

In the above example, the first index value i specifies a row index, whilst j specifies a column index.

DECLARATION

```
int m1[10][10];

static int m2[2][2] = { {0,1}, {2,3} };

sum = m1[i][j] + m2[k][l];
```

NOTE the strange way that the initial values have been assigned to the two-dimensional array m2. Inside the braces are,

{ 0, 1 },

{ 2, 3 }

Remember that arrays are split up into row and columns. The first is the row, the second is the column. Looking at the initial values assigned to m2, they are,

m2[0][0] = 0 m2[0][1] = 1 m2[1][0] = 2 m2[1][1] = 3

Example:

```
#include <stdio.h>

main()

{

static int m[][] = { {10,5,-3}, {9, 0, 0}, {32,20,1}, {0,0,8} };

int row, column, sum;

sum = 0;

for( row = 0; row < 4; row++ )

for( column = 0; column < 3; column++ ) sum = sum + m[row][column]; printf("The total is %d\n", sum );

}
```

CHARACTER ARRAYS [STRINGS]

Consider the following program,

```
#include <stdio.h>

main()

{

static char name1[] = {'H','e','l','l','o'}; static char name2[] = "Hello"; printf("%s\n", name1);

printf("%s\n", name2);
```

}

The difference between the two arrays is that name2 has a null placed at the end of the string, ie, in name2[5], whilst name1 has not. To insert a null at the end of the name1 array, the initialization can be changed to,

static char name1[] = {'H','e','l','l','o','\0'};

Consider the following program, which initialises the contents of the character based array word during the program, using the function strcpy, which necessitates using the include file string.h

Example:

#include <stdio.h>

#include <string.h>

main()

{

char word[20];

strcpy(word, "hi there.");

printf("%s\n", word);

}

---

3. Explain the standard string functions with example to support each type.

Strings:

The group of characters, digits and symbols enclosed within quotes is called as strings or character arrays. Strings are always terminated with '\0' character(NULL).

Example:

char name[ ] = {'H','E','L','L','O'};

Standard String Functions:

· strlen( )

· strcpy( )

· strncpy( )

· stricmp( )

· strcmp( )

· strncmp( )

· strcat( )

· strrev( ) etc.,

Example program:

· To read and display a string.

· Program to count the number of lines, words and characters in a text.

---

4. What are pointers? When and why they are used? Explain in detail with sample programs. (JAN 2009/MAY 2009)

Pointer variable is needed to store the memory address of any variable. Denoted by

(*) asterisk.

Pointer Declaration: Syntax:

datatype *variable-name;

Exmaple:

int *a;

· Pointers and Arrays

· Pointers and Strings

· Pointer as function arguments

· Pointer too pointer

Example program:

· To add two numbers through variables and their pointers.

· To assign a pointer value to another variable.

---

5. Describe in detail about the Preprocessors in C. (MAY 2009)

THE PREPROCESSOR

The define statement is used to make programs more readable, and allow the

inclusion of macros. Consider the following examples,

#define TRUE 1 /* Do not use a semi-colon , # must be first character on line */

#define FALSE 0

#define NULL 0

#define AND &

#define OR |

#define EQUALS ==

game_over = TRUE;

while(list_pointer != NULL)

.....

MACROS

Macros are inline code which are substituted at compile time. The definition of a macro, which accepts an argument when referenced,

#define SQUARE(x) (x)*(x)

y = SQUARE(v);

In this case, v is equated with x in the macro definition of square, so the variable y is assigned the square of v. The brackets in the macro definition of square are

necessary for correct evaluation.

The expansion of the macro becomes y = (v) * (v);

Naturally, macro definitions can also contain other macro definitions,

```
#define IS_LOWERCASE(x) (( (x)>='a') && ( (x) <='z') )

#define TO_UPPERCASE(x) (IS_LOWERCASE (x)?(x)-'a'+'A':(x))

while(*string) {

*string = TO_UPPERCASE (*string);

++string;

}
```

CONDITIONAL COMPILATIONS

These are used to direct the compiler to compile/or not compile the lines that

follow

```
#ifdef NULL

#define NL 10

#define SP 32

#endif
```

In the preceding case, the definition of NL and SP will only occur if NULL has

been defined prior to the compiler encountering the #ifdef NULL statement. The scope of a definition may be limited by

```
#undef NULL
```

This renders the identification of NULL invalid from that point onwards in the source file.

Typedef

This statement is used to classify existing C data types, eg,

```
typedef int counter; /* redefines counter as an integer */

counter j, n; /* counter now used to define j and n as integers */
```

```
typedef struct {

int month, day, year;

} DATE;

DATE todays_date; /* same as struct date todays_date */
```

ENUMERATED DATA TYPES

Enumerated data type variables can only assume values which have been previously

declared.

```
enum month { jan = 1, feb, mar, apr, may, jun, jul, aug, sep, oct, nov, dec };

enum month this_month;

this_month = feb;
```

In the above declaration, month is declared as an enumerated data type. It consists of a set of values, jan to dec. Numerically, jan is given the value 1, feb the value 2, and so on. The variable this_month is declared to be of the same type as month, then is assigned the value associated with feb. This_month cannot be assigned any values outside those specified in the initialization list for the declaration of month.

Example:

```
#include <stdio.h>

main()

{

char *pwest = "west",*pnorth = "north", *peast="east", *psouth = "south";

enum location { east=1, west=2, south=3, north=4};

enum location direction;

direction = east;

if( direction == east )

printf("Cannot go %s\n", peast);
```

}

The variables defined in the enumerated variable location should be assigned initial

values.

DECLARING VARIABLES TO BE REGISTER BASED

Some routines may be time or space critical. Variables can be defined as being register based by the following declaration,

register int index;

DECLARING VARIABLES TO BE EXTERNAL

Here variables may exist in separately compiled modules, and to declare that the

variable is external,

extern int move_number;

This means that the data storage for the variable move_number resides in another source

module, which will be linked with this module to form an executable program. In using a variable across a number of independently compiled modules, space should be allocated in only one module, whilst all other modules use the extern directive to access the variable.

NULL STATEMENTS

These are statements which do not have any body associated with them.

/* sums all integers in array a containing n elements and initializes */

/* two variables at the start of the for loop */

for(sum = 0, i = 0; i < n; sum += a[i++])

;

/* Copies characters from standard input to standard output until EOF is reached */

for(; (c = getchar ()) != EOF; putchar (c));

COMMAND LINE ARGUMENTS

It is possible to pass arguments to C programs when they are executed. The brackets which follow main are used for this purpose. argc refers to the number of arguments passed, and argv[] is a pointer array which points to each argument which is passed to main. A simple example follows, which checks to see if a single argument is supplied on the command line when the program is invoked.

```
#include <stdio.h>

main( int argc, char *argv[] )

{

if( argc == 2 )

printf("The argument supplied is %s\n", argv[1]);

else if( argc > 2 )

printf("Too many arguments supplied.\n");

else

printf("One argument expected.\n");

}
```

Note that *argv[0] is the name of the program invoked, which means that

*argv[1] is

a pointer to the first argument supplied, and *argv[n] is the last argument. If no arguments

are supplied, argc will be one. Thus for n arguments, argc will be equal to n + 1. The program is called by the command line, myprog argument1.

---

6. Brief call by value and call by reference in detail. (MAY 2009)

Call by value:

In call by value the value of the actual arguments are passed to the formal arguments and the operation is done on formal arguments.

Example program:

· To send two integer values using “call by value”.

Call by reference:

In call by reference the address of actual argument values are passed to formal argument values.

Example program:

· To send a value by reference to user defined function.

---

7. Discuss about function prototypes in detail. (or)

Explain about the different parameter passing methods with examples (JAN 2009)

· Function with arguments and return type.

· Function without arguments and return type.

· Function with arguments and no return type.

· Function without arguments and return type.

---

8. Define Structures. Explain structures in detail. (JAN 2009 / MAY 2009)

A structure is a collection of one or more variables of different data types grouped together under a single name. It contains different data types.

Syntax:

struct struct-name

{

type variable 1; type variable 2; type variable n;

} structure_variables;

Example:

struct student

{

char name[25];

int rollno;

int m1,m2,m3,total;

float avg;

}s1,s2;

· Structure within structure

· Array of structures

· Pointers to structures

· Structures and functions

Example program:

· To define a structure and read the member variable values from user.

· To copy structure elements from one object to another object.

---

9. Define Union. Explain Union in detail. (JAN 2009)

Union is a collection of variables similar to structure. The union requires bytes that are equal to number of bytes required for the largest number.

Example:

union student

{

char name[20];

int rollno,m1,m2,m3,tot;

float avg;

}s1;

Union of structure

Union can be nested with another union.

Example program:

· Program to use structure within union. Display the contents of structure elements