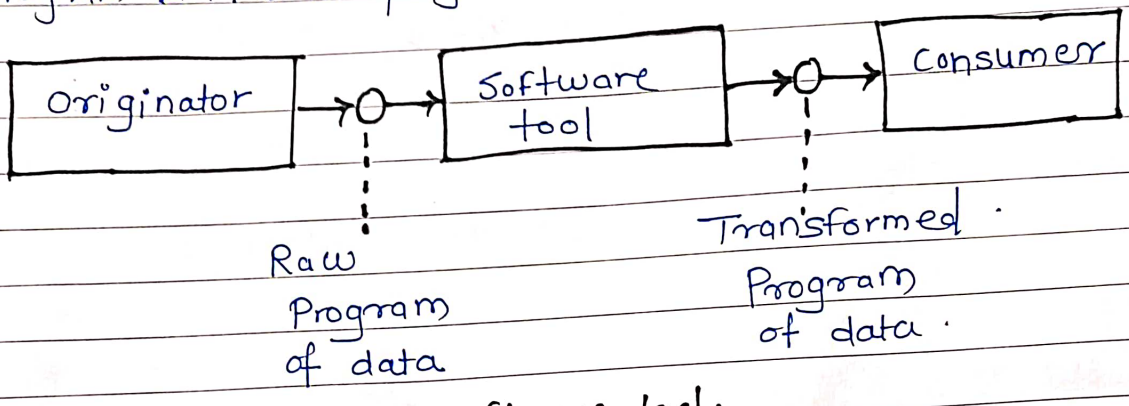
Software tool.

Software tool.

→ A software tool is a system program which
1- interface a program with the entity generating its input data or.
2- interfaces the results of a program with the entity consuming them.
- A software tool is a system program that suitably interface a program with other program a human users in its environm
-



Software tool.

Que1. Explain software tools for program development or discuss in details steps in program develop

⇒ The Fundamental steps in program development are
1- Program design, coding and documentation.
2- Preparation of programs in machine readable for
3- Program translation, linking and loading.
4- Program testing and debugging.
5- Performance tuning.
6- Source code management and versioning.
7- Reformatting the data and/or result of a progr to suit other programs.

① Program Design and coding.

Two categories of tools used in program design and codi are.
1- Program generators.
2- Programming environment.

A Program generator generates a program which Performs a set of functions described in its specification.
- use of a program generator saves substantial design effort.
- since a programmer merely specifies what functions a Program should perform rather than how the functions should be implemented.
- coding effort is also saved since the program is generated rather than coded by hand.

- **Programming environment :-**

Programming environment-supports program coding by incorporating awareness of the syntax and semantics of a programming language in an editor for the language.
- It use save substantial coding and testing effort

(2) - Preparation of Programs in machine readable form

- To Prepare Programs in machine readable form using a software tool, you typically write code in a programming language and use a text editor or integrated development environment (IDE).

- once the code is written, you save it in a file with the appropriate file extension for the programming language.

- Most programming languages have compilers or interpreters that process the human-readable code and generate machine-readable instruction.

- This compilation or interpretation step transforms the code into a format that the computer's hardware can execute

(3)

3. Popular software tools for code development include Visual Studio Code, IntelliJ IDEA, Eclipse, and many others, depending on the programming language you are using.

(3) Program translation, linking and loading.

(I) - Program Translation:-
- Compilation - The source code written by a programmer is translated into machine code or an intermediate code by a compiler.
- This process checks for syntax errors and produces an executable file.

(II) - Linking :-
i) Static linking :- In this process, the linker combines the compiled code with external libraries to create a single executable file.
- The linker resolves references between different parts of the program.

ii) Dynamic linking :- instead of combining everything into a single file, dynamic linking allows references to be resolved during runtime.
- Dynamic link libraries (DLLs) or shared libraries contain the code that can be linked at runtime.

(III) loading :-
- Loading into memory :- The operating system or a loader ~~loads~~ loads the executable file into memory.
- This involves allocating memory space for the program and its variables.

- Address Binding - The addresses used in the program are bound to specific memory locations during the loading phase. This can be done at compile time (static binding) or runtime (dynamic binding).

GoodLuck | Page No.
Date

④ · Program testing and debugging

- In system Programming, software testing & debugging are critical due to the low-level nature of the task involved.

1. Testing

(I) - Unit Testing :- verify individual functions, modules, or components of the system to ensure they operate correctly.

(II) - Integration testing :-
testing the data flow between two modules
- validate the interaction between different system modules or components.

(III) - System Testing :- Assess the entire system's functionality often involving scenarios that mimic real-world usage.

2 - Debugging

- Core Dumps and Crash Analysis
- system programming often deals with low-level languages like C or assembly, where debugging involves analyzing core dumps or crash reports to identify the cause of failures.

- Hardware interaction -
Debugging in system programming may involve dealing with hardware interaction, and tools like GDB (GNU debugger can be instrumental.

Imp
- Debug Monitors -
- Debug monitors help in obtaining information for localization of errors.
- localization & removal of errors has been aided by special purpose debug information.

(5) - Performance Tuning

- Efficiency of a program depends on two factors - efficiency of the algorithms used in it and efficiency of its code.
- An optimizing compiler can improve efficiency of the code but only a program designer can improve efficiency of the algorithm.

- both are time-consuming processes hence some help should be provided to improve their cost-effectiveness.

- It is observed that the less than 3% of a programs code generally consumes more than 50% of its execution time.

- Thus performance improvement efforts should be focused on only those parts of a program that consume a considerable amount of execution time.

*e.g- Program performance tuning.

execution times consumed by modules of a program.

Module	# of Statement	% of total execution time
A	150	4.00
B	80	6.00
C	35	90.00

- A program consists of module A, B and C.
It shows sizes of the modules & execution times consumed by them in a typical execution of the Program.
- Module C. Which is roughly 13% of the total program size, & consume 90% of its execution time.
- Hence optimization of module C would result in optimization for 90% of program execution time at only 13% of the cost.

Goodluck

GoodLuck | Page No.
Date

- A profile monitor is a software tool that collects information regarding the execution behaviour of a program.

e.g. - the amount of execution time consumed by its modules, & present it in the form of an execution profile.

- using this information the programmer can focus attention on parts of the program that consume a significant amount of execution time.

- These parts can be improved either through code optimization, or through improvement of the algorithm

(6) Source code management and version control.

- A program goes through many modifications while it is being developed and used.
- The first few modifications occur during debugging.
- Each of these modification is based on a hypothesis about the cause of a bug & how it can be corrected.
- The program is tested once again after making the modification.
- If the bug persists, the modification must be undone and fresh hypothesis and modication must ~~to~~ follow. This process is repeated until the program is considered to be bug-free.
- A modification involves changes in many statemen and insertion & deletion of statements.
- Hence. Undoing of a modification is a complex task that cannot be achieved by simply using the undo facility of an editor.

GoodLuck | Page No.
Date / /

- The task become more complex if modules of a Program are stored in different files.
- Hence undoing of a modification involves undoing of all changes in the batch.
- If some of the changes in a batch are undone but others are not, the resulting program would be inconsistent,
- Which would affect further testing & modification of the program.
- such inconsistencies may aries due to negligence or due to a system crash or power failure during the undoing of changes.
- A source code management system is a software tool that uses special techniques to implement making and undoing of program modifications in a consistent manner.

(7) - Reformatting the data and result of a program to suit other programs.

to reformat data and program results for compatibility with another program, consider the following steps.

1- understand ~~data~~ data structures:-

- Analyze the data structures used in both programs.
- Ensure compatibility by mapping equivalent structures.

2- Serialization formats:-

- choose a suitable serialization format (JSON, XML. etc) for data interchange.
- serialize data in the sending program & deserialize in the receiving program.

3- Communication protocols :-

- Establish clear communication protocols between the programs.
- Define how data will be transmitted, received and interpreted.

4- Standardize Interfaces :-
- Ensure that the input & output interfaces align between the programs.
- use common conventions for data representation.

5- Testing and validation :-
- Rigorously test data exchange between the programs.
- verify that reformatted data maintains integrity and consistency

6- Documentation -
- Document the data formats and communication protocols for future reference.
- provide clear instructions for any future developers working on the integration.

Que 2 - What is Editor ? explain structure of Editor with suitable Diagram.

or

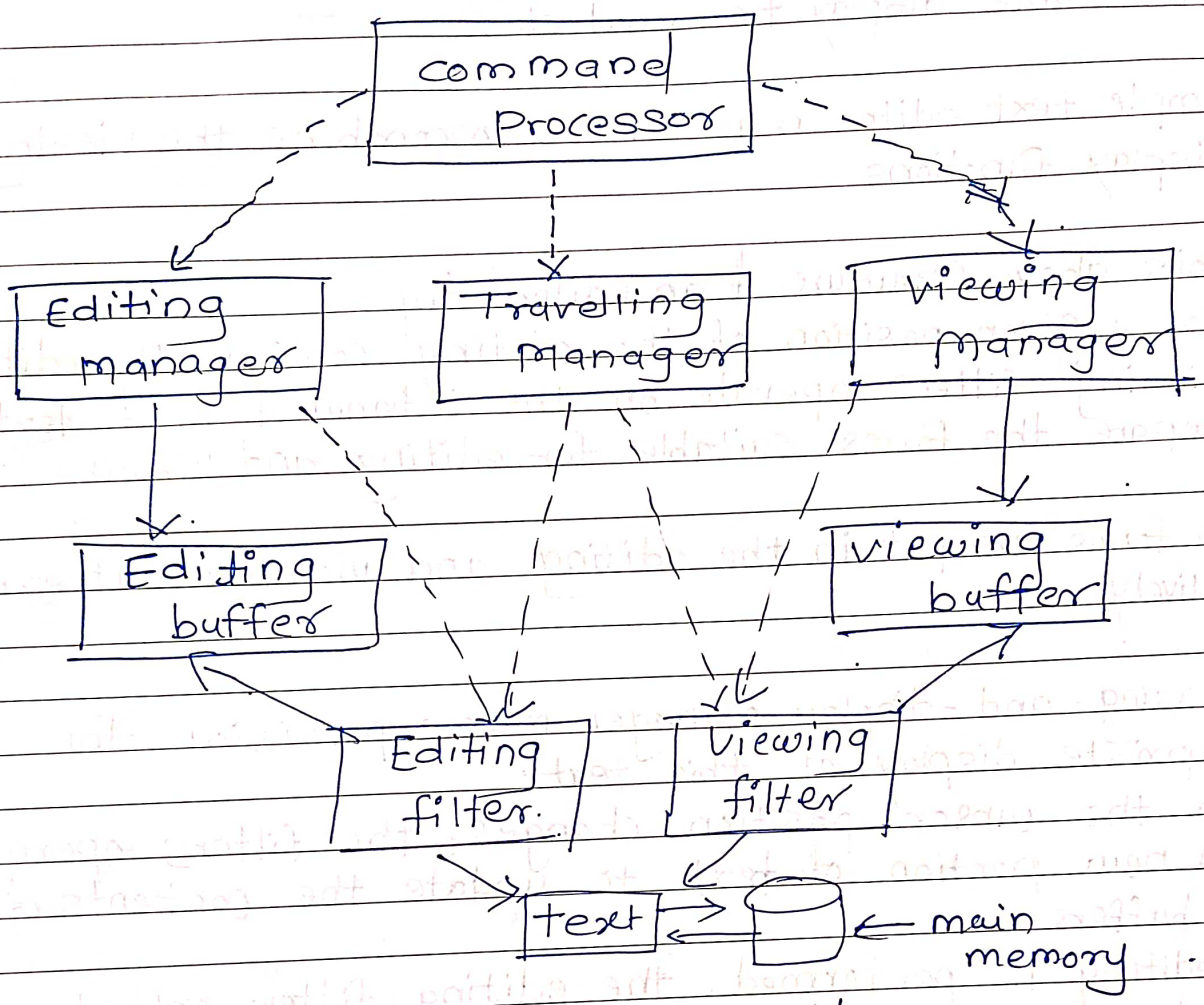
Explain Design of an editor with suitable example.

Ans :- The fundamental functions in editing are travelling, editing, viewing & display.

- Travelling implies movement of the editing context to a new position within the text.
- It may be done either explicitly by the user command (eg - the line number command of line editor).

GoodLuck | Page No.
Date

or- it may be implied in a user command. e.g- the search command of a stream editor.



structure of an editor.

- Viewing implies formatting the text in a manner desired by the user.
- It provides an abstract view. independent of the physical characteristics of an IO (input output) devices.
- The display component maps this view into the physical characteristics of the display device being used.
- It determines where a particular view may appear on the user's screen.

- The separation of viewing and display functions gives rise to interesting possibilities like multiple windows on the same screen, concurrent edit ~~interesting~~ operations using the same display terminal, etc.

- A simple text editor may choose to combine the viewing and display functions.

In this above (structure of an editor) fig.
for a given position of the editing context, the editing and viewing filters operate on the internal form of text to prepare the forms suitable for editing and viewing.

- These forms are put in the editing and viewing buffers respectively.

- The viewing-and-display manager makes provision for appropriate display of this text.
- When the cursor position changes, the filters operate on a new portion of text to update the contents of the buffers.
- As editing is performed, the editing filter reflects the changes into the internal form and updates the contents of the viewing buffer.

- A part from the fundamental editing functions, most editors support an undo function to nullify one or more of the previous edit operations performed by the user.

- The undo function can be implemented by storing a stack of previous views or by devising an inverse for each edit operation.

- multilevel undo commands pose obvious difficulties in implementing overlapping edits.

Goodluck | Page No.
Date

que 3- Explain Types of editors with an example for each editor.

Editor - An editor is computers program that allows a user to create & modify a target document or file.

Types of editor

1] Line editor
2] steam editor
3] screen editor
4] word processor editor.
5] structure editor

① Line editor. - Editing operands in the line editor is performed on a line of text every line in a text document is identified either in one of two ways.

i] positionally - i.e- can be use as line no. in text.
ii] contextually - i.e. using context which uniquely identify types.

3]- line editor has to maintain multiple representation of text.

i] Display form - It shows the text as a sequence of line used to view text on screen.

ii] Internal form- It is used to perform edit operation it contain for end line ⇒ edit character.

Advantages- simple to develop & design.

disadvantages- we can not perform edition on part of line.

ii) It can not display, text as it appears if printed
e.g - editor 'ed'
mean. you have to give input in horizontally but when you are printed this the input will be vertically not as it is horizontally i.e- It can not display text as it appears if printed.

2] • Stream editor –

- This editor views in either text as a shows of characters.

2] - It performs edit operation to cross line boundries. (means multiple lines)

3] support character line as well as context oriented commands

4] - similar to line each editor also maintain multiple representation of text - i] Display form ii] Internal form

e.g - 'sed' in unix.

Advantages - It can performed operation on part of line as well as can cross line boundries.

Disadvantages

- It is difficult to develop & design not userfriendly
- It does not display text / in the manner it appear

3] • Screen Editor

- What - You - See - Is - What - You - GET principal is used.

- It display the text in the mannar it would appear.
- Display a complete text on screen & user can more cursor over it. (editing operation like cut, copy, paste delete).
- user can position cursor to operation
- Therfore user can see editing on Sureen.

- A line or Steam editor does not display the text in the manner it would appear if printed.

Goodluck

- The editor displays a screenful of text at a time.
- The user can move the cursor over the screen, position it at the point where he desires to perform some editing & proceed with the editing directly
- Thus the effect of an edit operation can be seen on the screen immediately. This feature is very useful while formatting the text to produce printed documents

Advantages :-

- More interactive & user friendly
- Editing effects are immediatly visible on screen
- uses what-you-see-is-what-you-get principal so text appear on the screen is same format as it appear on print.

Disadvantages :-

- Difficult to design & develop as compare to line & editor (steam).

• 4] Word processor

- Word processors are basically document editors with additional features for producing well formatted hard copy output.
- Essential features of word processors are commands for moving sections of text from one place to another.
- Merging of text and searching & replacement of words.
- Many word processors support a spell-check option with the advent of personal computers, word processors have seen widespread use amongst authors, office personnel & computer professionals.
- Microsoft word is a popular editor of this class.

Goodluck

Advantages

- More user friendly
- provide GUI base platform.
  [Graphics user interface].

Disadvantages-
- It suitable for document editing & not for editing in computer program.

• 5) Structure editors.

- A structure editor incorporates an awareness of the structure of a document.

- This awareness is useful in browsing through a document,
  e.g - if a programmer wishes to edit a specific function in a program file.

- The structure is specified by using the structure.

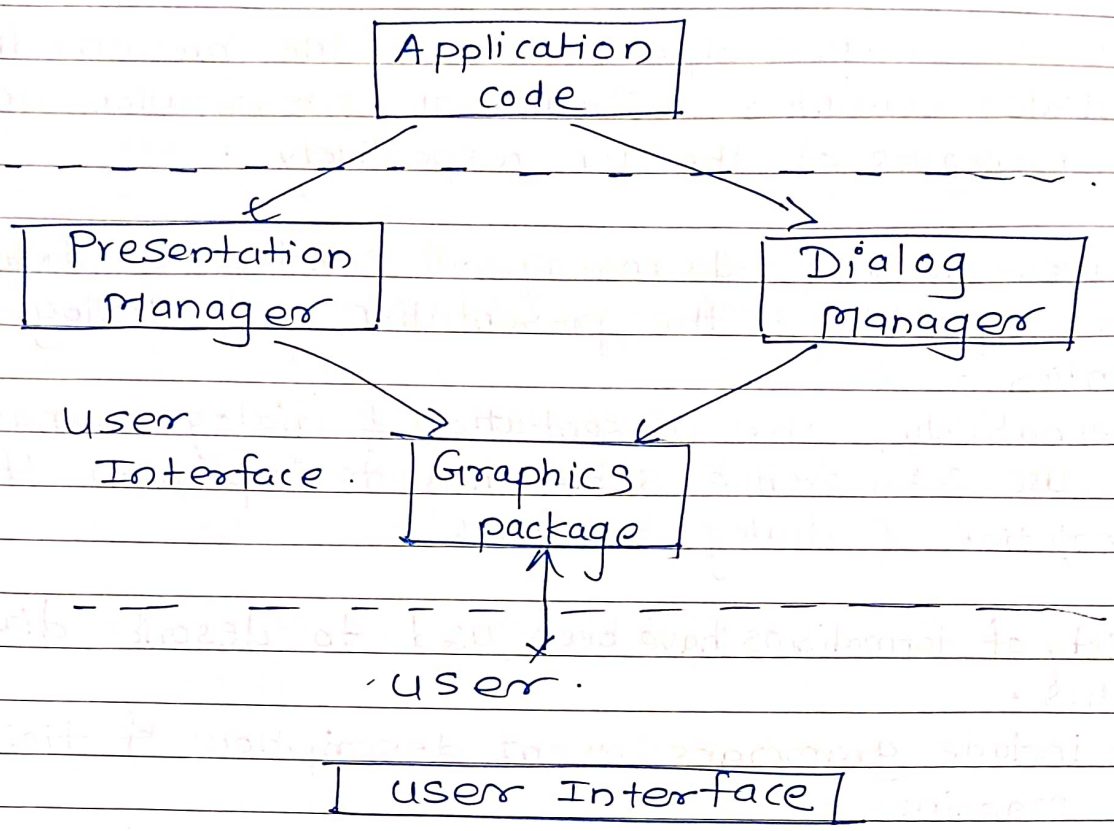
- A special class of structure editors, called syntax directed editors, are used in programming environments.

Goodluck

GoodLuck | Page No. | Date

Que-4] - Explain structure user Interface?

Ans :-



User Interface

- UI schematic using a standard graphics packag
- The UI consists of two main components - the presentation manager & the dialog manager.
- The presentation manager is responsible for managing the user's screen and for accepting data & presenting result.
- The dialog manager is responsible for interpreting user commands and implementing them by invoking different modules of the application code.
- The dialog manager is also responsible for error messages and on-line help functions, and for organizing changes in the visual context of the user.
- A user interface management system (UIMS) automate the generation of user interfaces.

- The UIMS accepts specification of the presentation and dialog semantics to produce the presentation and dialog managers of the UI respectively.

- The presentation & data managers could be generated programs specific to the presentation and dialog semantics.
  - Alternatively, the presentation & dialog managers could use interpretive schematics to implement the presentation & dialog semantics.

- A variety of formalisms have been used to describe dialog semantics.
- These include grammars, event descriptions & finite state machines.
- In a grammar based description, the syntax & semantics of commands are specified in a YACC like manner.
  - Thus, the interface is activated when a user types in a command.
  - The event based approach uses a visual model of the interface.
- A screen with icons is displayed to the user. Selection of an icon by clicking the mouse on it causes an event
  - The action specified against the event is now performed.

- The grammar & event description approaches lack the notion of a sequence of action.
- The finite state machine approach can efficiently incorporate this notion.
- The basic principle in this approach is to associate a finite state machine with each window or each icon.
- Actions are specified on the basis of conditions involving the state of these machines.
- This approach has the additional power of coordinating the concurrent activities in different windows.

• two UIMSs using the event description approach.

(I) Menuay :-
- Menuay is an early UIMS using the screen layout as the basis for the dialog model.
- The UI designer starts by designing the user screen to consist of a set of icons.
- A semantic action is specified for each icon. This action is performed when the icon is selected.
- The interface consists of a set of screens.

- The system generates a set of icon tables giving the name & description of an icon, & a list of (event, function_id) pairs indicating which application function should be called when an event is selected.

(II) Hypercard -
- This UIMS from Apple incorporates object orientation in the event oriented approach.
- A card has an associated screen layout containing buttons & fields.
- A field contains editable text. each card has a specific background, which itself behaves like a card. many cards can share the same background.

- A hypercard program is thus a hierarchy of cards called a stack.

- UI behaviour is specified by associating an action, in the form of HyperTalk script, with each button, field & card.

- The action for an event is determined by using the hierarchy of cards as an inheritance hierarchy.
- Hypercard uses an interpretive schematic to implement a UI

Goodluck

.5] short notes.

i) Debug Monitor.

Debug Monitors provide the following facilities for dynamic debugging:

1. setting breakpoints in the program.
2. Initiating a debug conversation when control reaches a breakpoint.
3. Displaying values of variables
4. Assigning new values to variables.
5. Testing user defined assertions & predicates involving program variables.

ii) The debug monitor functions can be easily implemented in an interpreter. However interpretation incurs considerable execution time penalties.

iii) A debug monitor therefore relies an instrumentation of a compiled program to implement its function.

iv) To enable the use of a debug monitor, the user must compile the program under the debug option.

v) The compiler now inserts a few no-op instructions of the form:

no-op <statement no>

before each statement, where <statement no> is a constant indicating the serial number of the statement in the program.

vi.] - The compiler also generates a table containin the pairs (variable name, address).
- When a user gives a command to set a breakpoint at, say, statement 100, the debug monitor instruments the program to insert the instruction

<SI_instrn> <code>

- in place of the no-op instruction no-op 100.
- The compiled code for the program executes directly on the CPU until it reaches an <SI_instrn>.
- Execution of the <SI_instrn> produces a Software interrupt with the interrupt code <code>, which signifies a debug interrupt.

(Alternatively, the debug monitor can use the instruction BC ANY, DEBUG_MON insted of <SI_instrn>).

- The debug Monitor now gains control & opens a debug conversation.

- The user may ask for display or modification of program variables.

- It is implemented by using the (variable name, address) information produced by the compile

Imp

- The sequence of steps involved in dynamic debugging of a program is as follows.

1. The User compiles the program under the debug option. The compiler produces two files - the compiled code file & the debug information file

2. The User activates the debug monitor & indicates the name of the program to be debugged.
   - The debug monitor opens the compiled code & debug information files for the program.

3. The User specifies his debug requirements - A list of breakpoints & actions to be performed at breakpoints
   - The debug monitor instruments the program and builds a debug table containing the pairs (Statement number, debug action)

4. The instrumented program gets control and executes up to a breakpoint.

5. A software interrupt is generated when the <SI_instrn> is executed.
   - control is given to the debug monitor which consults the debug table & performs the debug actions specified for the breakpoint.

6. - A debug conversion is now opened during which the user may issue some debug commands. (Which are implemented through interpretation)

or modify breakpoint & debug actions associated with the breakpoints
- control now return to the instrumented program

6- Step 4 & 5 are repeated until the end of the debug session

- Unix supports two debuggers - sdb which is a source-level debugger & adb which is an assembly language-level debugger.
- GDB of the GNU project provides debugging for many programming languages under unix & Linux

GNU stands for "Gnu's Not Unix"
GDB stands for "Gnu DeBugger"

- short notes.

ii) User - Interface.

- A user interface (UI) plays a vital role in simplifying the interaction of a user with an application.

- UI functionalities have two important aspects
  i) issuing of commands & exchange of data.
  in early days of computing, a user was often the application designer or developer.

  ii) understanding of commands & data was implicit in the use of a application.
  in those days UI's did not have an independent identity.

- This situation changed because of two reasons.
  i) - as applications became larger a user was no longer expected to know all details concerning an application.
  Hence - presentation of commands & prompts for data became important

  ii) - as applications grew to newer fields, it became necessary to assume a lower level of computer skills is an application user.

- It increased the importance of UI's by adding a new aspects, viz. on-line help, to their functionality.

The on-line help component of the UI serves the function of educating the user in the capabilities of an application & the modalities of its usage.

- A UI can be visualized to consist of two components -
- A dialog manager & a presentation manager.

- The dialog manager manages the conversation between the user & the application
- It involves prompting the user for a command & transmitting the command to the application
- The presentation manager displays the data produced by the application in an appropriate manner on the user's display or printer device.

Q 5. Define debug monitor.

A debug monitor is a software tool or component used in the process of debugging computer programs. It allows developers to monitor & analyze the execution of a program.

- helping them identify & correct errors or unexpected behavior.

Key features of debug monitor may include

1- Breakpoint

- The ability to pause the execution of a program at specified points, allowing developers to inspect the program's state & variables.

2- Stepping

- The capability to execute a program one line or one instruction at a time, aiding in the detailed examination of code execution.

3. Variable Inspection -

- Tools for examining the values of variables during runtime, helping developers understand how data charges as the program runs.

4. Memory Inspection

- The ability to inspect and modify the contents of memory locations, crucial for identifying memory-related issues.

5. Call stack Examination:

Viewing the call stack to understand the sequence of function calls leading to a specific point in the program.

6. output Monitoring- capturing & displaying program output, error messages, or other relevant information during execution.

- Debug monitors are essential for the development & troubleshooting process, especially in complex software projects.

- They come integrated into integrated development environments (IDEs) or can be standalone tools used alongside a compiler & debugger.

ue-6- Explain about the tools used in Enhancement of program performance? / programming environments?

ns- A programming environment is a software system that provides integrated facilities for program creation, editing, execution, testing & debugging.

- Its use avoids use of distinct software tools during different steps in program development.

- thereby providing convenience & enhancing programmer productivity, A software environment consist of the following components.

1- A syntax directed editor (which is a structure editor

2- A language processor - A compiler, interpreter or both

3- A debug monitor

4- A dialog monitor.

- All components are accessed through the dialog monitor.

- The syntax directed editor incorporates a front end for the programming language.

- As a user keys in his program, the editor performs syntax analysis & convert it into an intermediate representation, typically an abstract syntax tree
- The compiler (or interpreter) & the debug monitor share the intermediate representation
- If a compiler is used, it is activated after the editor has converted a statement to intermediate representation
- The compiler works incrementally to generate code for the statement
- This way, execution or interpretation of the program can start immediately after the last statement has been input.
- The programmer can interrupt execution of the program at any time to either enter & debug mode or return to the editor, modify the program & resume or restart its execution
- The main simplification for the user is the easy accessibility of all functions through the dialog monitor.

The system may also provide other program development & testing functions.

for e.g - it may permit a programmer to execute a partially completed program.

- The programmer can be alerted if an undeclared variable or an incompleted statement is encountered during execution.

- The programmer can insert necessary declaration or statements & resume execution.

- This arrangement permits major interfaces in the program to be tested prior to the development of a module.

Some programming environment also support reversible execution,

whereby a program's execution can be 'stepped back' by one or more statements

i) Debug Monitor.

Debug monitors provide the following facilities for dynamic debugging:

1. setting breakpoints in the program.
2- Initiating a debug conversation when control reaches a breakpoint.
3- Displaying values of variables.
4- Assigning new values to variables.
5- Testing user defined assertions & predicat involving program variables.

ii) - The debug monitor functions can be easily implemented in an interpreter. However interp interpretation incurs considerable execution tim penalties.

iii) A debug monitor therefore relies an instrumenta of a compiled program to implement its functio

iv) To enable the use of a debug monitor, the user must compile the program under the debu option.

v) The compiler now inserts a few no-op instruc instructions of the form:

no-op <statement no>

before each statement, where <statement no> is a con indicating the serial number of the statement in

Vi.] - The compiler also generates a table containing the pairs (variable name, address).

- When a user gives a command to set a breakpoint at, say, statement 100, the debug monitor instruments the program to insert the instruction:

<SI_instrn> <code>

- in place of the no-op instruction no-op 100.
- The compiled code for the program executes directly on the CPU until it reaches an <SI_instrn>.
- Execution of the <SI_instrn> produces a Software interrupt with the interrupt code <code>, which signifies a debug interrupt.

(Alternatively, the debug monitor can use the instruction BC ANY, DEBUG_MON instead of <SI_instrn>).

- The debug monitor now gains control & opens a debug conversation.

- The user may ask for display or modification of program variables.

- It is implemented by using the (variable name, address) information produced by the compiler.

Q.7] What is command Dialog? Explain ways to implement command Dialog.

→ Commands are issued to an application through a command dialog.

- A command dialog generally refers to a communication process where commands or instructions are exchanged between a user and a system.

- In the context of software or computing, it often involves users providing commands to a program or system, & the system responding accordingly.

- This can be through a command-line interface, voice commands, or other means of input depending on the technology in use.

- It can be implemented in the following three ways.

1. Command languages
2. command menus
3. Direct manipulation.

i) command languages for computer applications are similar to command languages for operating systems.

primitive command languages support imperative commands with the syntax.

<action> <parameters>.

more sophisticated command languages have both declarative & imperative commands & a syntax & semantics of their own.

- A practical difficulty in the use of a command languages is the need to learn it before using the application.

- It implies a large commitment of time & effort by the user, which makes casual use of the application difficult.

- on-line help can provide some relief by avoiding the need to memorize the syntax of commands.

- however it cannot eliminate the need to invest time & effort in initial learning of the command languages.

- Command menus

- command menus provide obvious advantages to the casual user of an application, as the basic functionalities of the application are reflected in the menu choices.

- A hierarchy of menus can be used to guide the user into the details concerning a functionality, interesting variations like pull-down menus are designed to simplify the use of menu systems.

Direct Manipulation:-

- A direct manipulation system provides the user with a visual display of the universe of the application.

- The display show the important objects in the universe actions or operations over objects are indicated by using a pointing device like a cursor or a mouse.

e.g. of common dialog through direct manipulation.

- A spreadsheet supports direct manipulation.

- The universe of the application is the spreadsheet. The user can select a specific cell in the spreadsheet, change its content & study the effect of this change on other cells.

e.g. of direct manipulation can also be found in screen editors & ~~tied~~ video games.

principles of command dialog design.

1- easy to use
2- consistency in command structure
3- Immediate feedback on user commands
4- error handling
5- on-line help avoid memorizing command details
6- Undo facility
7- Shortcuts for experienced users.