SDM College of Engineering and Technology

Dhavalagiri, Dharwad-580 002. Karnataka State. India.

Email: principal@sdmcet.ac.in, cse.sdmcet@gmail.com

Ph: 0836-2447465/ 2448327 Fax: 0836-2464638 Website: sdmcet.ac.in

Department of COMPUTER SCIENCE AND ENGINEERING

MINOR WORK REPORT

[22UHC500- Software Engineering and Project Management]

Odd Semester: September 2024-January 2025

Course Teacher: Dr. U.P.Kulkarni

2024- 2025

Submitted by
By

Varsha S.B

2SD22CS122

5th Semester B division

Table of Contents

1. ASSIGNMENT-1 3

1.1 Problem Statement 3

1.2 Theory 3

1.3 Program 3

1.4 Sample input and output 4

1.5 References 4

2. ASSIGNMENT-2 5

2.1 Problem Statement 5

2.2 Theory 5

2.3 Design 5

2.4 References 8

3. ASSIGNMENT-3 8

3.1 Problem Statement 8

3.2 Theory 9

3.3 Program 9

3.4 Sample input and output 11

3.5 References 11

4. ASSIGNMENT-4 11

4.1 Problem Statement 12

4.2 Theory 12

4.3 Program 12

4.4 Sample input and output 13

4.5 References 13

ASSIGNMENT-1

1.1 Problem Statement:

Write a C program to show that C programming Language supports only Call by Value.

Business Scenario:

Imagine a bank application where we are attempting to update the balance of an account. However, the function receives only a copy of the original balance and not the actual value, so the real balance remains unchanged outside of the function.Here's a C program demonstrating that C supports call by value, meaning when arguments are passed to a function, the function operates on copies of the actual arguments rather than the originals.

1.2 Theory:

Overview of Function Calling Mechanisms

- Call by Value: A copy of the actual parameter is passed to the function.
- Call by Reference: The actual parameter itself (memory reference) is passed to the function. During Call by Value when a function is called in C, copies of the arguments are created and used within the function. Any changes made to the parameters inside the function do not affect the original variables.

1.3 Program:

C Program: Bank Balance Update (Demonstrating Call by Value)

```
#include<stdio.h>
// Function to update the balance (Call by Value)
void updateBalance(int balance)
{
    // Adding $500 to the balance (Attempt to update balance)
    balance += 500;
printf("Inside updateBalance: Updated Balance = $%d\n", balance);
}
int main() {
```

```
    // Initial balance of the account
 printf("Enter the Initial Balance of the account");
 scanf("%d\n", balance);
    // Print initial balance
    printf("Initial Balance = $%d\n", balance);
    // Call function to update the balance
    updateBalance(balance);
    // Print balance after calling updateBalance
    printf("After updateBalance function call: Balance = $%d\n", balance);
    return 0;
}
```

1.4 Sample input and output:

Input: Enter the Initial Balance of the account =1000

Output: Initial Balance = $1000

Inside updateBalance: Updated Balance = $1500

After updateBalance function call: Balance = $1000

1.5 References:

1) www.GeeksforGeeks.com
2) Kernighan, Brain W., and Dennis M. Ritchie. The C Programming Language, 2nd Edition.

ASSIGNMENT-2

2.1 Problem Statement:

Study the concept of USABILITY. Prepare a report on USABILITY of at least two UIs of major software products you have seen.

2.2 Theory:

Usability Report on Major Software Product UIs: A Business Scenario Approach.The usability of software products is crucial for user satisfaction, efficiency, and retention. Below is an analysis of the usability of two major software UIs based on principles of ease of use, navigation, learnability, and aesthetics.

- **Effectiveness:** Measures how well users can complete tasks using the system. A highly effective system allows users to achieve their goals accurately and with minimal effort.
- **Efficiency:** Refers to the speed and resourcefulness with which users can complete tasks. An efficient system reduces unnecessary steps or redundant interactions, allowing for quicker task completion.
- **Error Tolerance:** Ensures that the system minimizes user errors and, when errors occur, provides clear instructions or pathways to recovery. This includes clear error messages and features like undo or auto-save.
- **Satisfaction:** Represents the overall pleasantness and fulfillment users experience. A satisfying design is aesthetically pleasing, free of frustrations, and meets user expectations.
- **Ease of use:** how easy it is for users to learn and begin using the system. Systems with high learnability offer intuitive navigation, clear instructions, and simple interfaces, enabling users to start quickly without a steep learning curve.

2.3 Design:

The design focuses on being a one-stop platform for workplace collaboration, while its flexibility with add-ons and customization has made it popular for diverse workflows and teams across industries.

Two Popular Software interfaces are:

1. Microsoft Teams is designed as a digital workspace that focuses on collaboration, communication, and integration with Microsoft 365 tools. Here are the main elements of its design:

Microsoft Teams is designed as a digital workspace that focuses on collaboration, communication, and integration with Microsoft 365 tools. Here are the main elements of its design:

- **User Interface (UI):**

Navigation: Teams uses a left-side vertical toolbar that provides quick access to core features like Chat, Teams, Calendar, Calls, and Files.

Channel Structure: Within a Team, channels are organized in a hierarchical structure that lets users focus on specific topics or projects.

- **Communication Tools:**

Chat and Channels: Teams integrate real-time chat, with both private and group chats, and threaded channel messages to keep conversations organized.

Audio/Video Calls: Supports voice and video conferencing with features like screen sharing, virtual backgrounds, and breakout rooms.

Meeting Enhancements: Tools like live reactions, real-time transcripts, meeting notes, and whiteboarding enhance collaborative sessions.

- **Integration with Microsoft 365 and Other Apps:**

Microsoft 365 Integration: Teams is closely integrated with other Microsoft tools like Outlook, Word, Excel, and PowerPoint. Files are stored in OneDrive and SharePoint, enabling seamless document collaboration.

Third-Party Apps: Teams allows users to integrate external apps (like Trello, Asana, or Zoom) as tabs or bots within channels and chats.

Automation and Bots: Teams offer bots for automating tasks, responding to commands, and providing reminders within channels.

- **Memorability**

Determines how easily returning users can re-establish proficiency. Systems that follow consistent design patterns and logical structures help users remember functions, even after a period of non-use.

Access Control: Teams provide detailed controls for admin roles, guest access, and information protection.

- **Collaboration and Productivity Tools:**

Document Collaboration: Files shared within Teams are automatically stored in SharePoint and are accessible for real-time co-editing.

Planner and To-Do: Microsoft Planner and To-Do are often integrated into Teams for task management and tracking project progress.

2. Adobe Photoshop is one of the most popular and powerful software tools for photo editing, graphic design, and digital art creation. Developed by Adobe, Photoshop offers an extensive range of tools for manipulating images, creating illustrations, and even designing user interfaces or animations.

❖ **Efficiency**

Adobe Photoshop is highly efficient for a range of design and editing tasks, thanks to its powerful tools and customization options. Photoshop provides a wide array of tools for photo manipulation, compositing, digital painting, and vector graphics. The range of features, like layers, masks, blending modes, and advanced brushes, makes complex editing more manageable and precise.

❖ **Effectiveness**

Adobe Photoshop is highly effective in various design and editing applications due to its powerful tools and flexibility. Photoshop is renowned for its advanced photo manipulation tools, including color correction, retouching, and layering, allowing precise adjustments to every element of an image.

❖ **Error Tolerance**

Adobe Photoshop incorporates several features to ensure error tolerance, allowing users to undo mistakes, recover work after crashes, and avoid unintentional edits. Key error tolerance features in Photoshop include.

Undo and Redo Functions: Photoshop offers multiple undo levels, accessible via Ctrl+Z (or Cmd+Z on Mac) and the History Panel, where users can move back and forth between steps. This allows recovery from recent errors and experimentation without permanent changes.

❖ **UI Interface**

Photoshop's workspace is modular, with customizable panels and toolbars that can be arranged to fit different workflows. Users can save and switch between workspace layouts, such as Photography, Graphic and Web, and Painting, each with tailored tool panels.

Located typically on the left side, the Tools Panel is one of the core elements, containing essential tools for selecting, drawing, retouching, and manipulating images.

Some popular tools here include the Move Tool, Marquee Tools, Lasso Tool, Brush Tool, Eraser Tool, and Clone Stamp Tool.

❖ **User Satisfaction**

Adobe Photoshop generally receives high satisfaction ratings due to its powerful image editing capabilities and vast array of tools that cater to both beginners and professionals. Photoshop offers numerous features, including layer-based editing, filters, brushes, retouching tools, and more, allowing users to create and edit images in great detail.

Professional Quality: Photoshop produces professional-grade results, often making it the standard software for image editing in industries like photography, graphic design, and digital marketing.

2.4 References:

1) Lean UX: Designing Great Products with Agile Teams, Jeff Gothelf and Josh Seiden.
2) The Elements of User Experience, Jesse Jamees Garrett.

ASSIGNMENT-3

3.1 Problem Statement:

List all the features of the programming language and write PROGRAMS to show how they help to write ROBUST code.

3.2 Theory:

Python is a versatile and widely-used programming language that is known for its simplicity and readability. Python is an incredibly versatile language with an extensive set of features that make it ideal for a wide range of applications, from web development and data analysis to AI and scientific computing.

Here is an extensive overview of the key features of Python:

1.Simple and easy to learn :

2.Exception Handling:

3. Interpreted Language:

4.High-Level Language:

5.Object-Oriented Programming (OOP) Support:

3.3 Program:

1.#A Simple Function to add two numbers in python

```
def add_numbers(a, b):
    return a + b
num1 = int(input("Enter first number: "))
num2 = int(input("Enter second number: "))
result = add_numbers(num1, num2)
print(f"The sum of {num1} and {num2} is: {result}");
```

2.#Here's a simple Python program demonstrating exception handling using try, except, and #finally blocks:

```
try:
number = int(input("Enter a number: "))
        result = 10 / number
        print(f"Result: {result}")
except ValueError:
        # This block handles invalid inputs that cannot be converted to an integer
        print("Error: Please enter a valid integer.")
except ZeroDivisionError:
        # This block handles division by zero errors
        print("Error: Division by zero is not allowed.")
finally:
        print("Execution completed.")
```

3.# A python program to show that it is executed line by line

```
print("Step 1: Program starts")
x = 10
print(f"Step 2: Variable x is set to {x}")
```

```
y = x + 5
print(f"Step 3: Variable y is set to {y}")
if y > 10:
        print("Step 4: y is greater than 10")
print("Step 5: Program ends")
```

4.#A simple program to add two numbers in python

```
print(3+4)
```

5. # OOP Concept: Class and Object

```
class Animal:
    def _init_(self, name):
    # OOP Concept: Encapsulation (data hiding)
    self.name = name
    # OOP Concept: Method (behavior of the object)
    def speak(self)
    raise NotImplementedError("Subclass must implement abstract method")
# OOP Concept: Inheritance
class Dog(Animal):
     def speak(self):
     # OOP Concept: Polymorphism (method overriding)
     return f"{self.name} says Woof!"
class Cat(Animal):
     def speak(self):
     return f"{self.name} says Meow!"
# Creating instances (objects) of classes
dog = Dog("Bob")
cat = Cat("Kitten")
```

```
# Calling methods of objects
print(dog.speak())
print(cat.speak())
```

3.4 Sample input and output:

1. Input: Enter first number = 2
   Enter second number = 5
   Output: The sum of 2 and 5 is 7
2. Input: Enter a number = 5
   Output:2.0
   Execution completed.
3. Output: Step 1: Program starts
   Step 2: Variable x is set to 10
   Step 3: Variable y is set to 15
   Step 4: y is greater than 10
   Step 5: Program ends
4. Output: 7
5. Output: Bob says Woof! Kitten says Meow!

3.5 References:

1) Real Python

2) Python.org

ASSIGNMENT-4

4.1 Problem Statement:

Study the "ASSERTIONS" in C language and its importance in writing RELIABLE CODE. Study POSIX standard and write a C program under Unix to show use of POSIX standard in writing portable code.

4.2 Theory:

- An assertion in C is a debugging tool used to test assumptions made by the program at runtime. It allows developers to specify a condition (usually an expression) that they believe should be true at a particular point in the program. If the condition evaluates to true, the program continues execution normally. If the condition evaluates to false, the program prints an error message and usually terminates the execution.In C, assertions are implemented through the assert() macro.

1. Importance of Assertions in Writing Reliable Code

    a. Identifying Logical Errors Early

    b. Defensive Programming

    c. Improving Code Robustness

    d. Simplified Debugging

- POSIX (Portable Operating System Interface) in the C language refers to a set of standardized APIs that make it easier to write programs that can run across different UNIX-like operating systems, including Linux and macOS. POSIX provides standards for several aspects of programming, such as file handling, process control, threading, and inter-process communication, allowing for code portability and consistency across compliant systems.

4.3 Program:

Here's a simple C program that demonstrates the use of POSIX standards to write portable code for Unix-like systems. The program retrieves the process ID (PID) and user ID (UID) using POSIX-compliant functions:

```
#include <stdio.h>

#include  <unistd.h>

#include <sys/types.h>

int main()
```

```
{
 // Get the process ID (POSIX-compliant function)
 pid_t process_id = getpid();
 // Get the user ID (POSIX-compliant function)
 uid_t user_id = getuid();
 // Display the process ID and user ID
 printf("Process ID: %d\n", process_id);
 printf("User ID: %d\n", user_id);
return 0;
}
```

This example ensures that the code is portable across Unix-like operating systems, as it uses POSIX-compliant functions.

4.4 Sample input and output:

Output:

Process ID: 12345

User ID: 1000

4.5 References:

1) Unix Programming , The Linux Documentation Project.
2) Unix Tutorials-GeeksforGeeks.