SDM COLLEGE OF ENGINEERING AND TECHNOLOGY,DHARWAD

Email: cse.sdmcet@gmail.com

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

A Report on

Course Work for CTA

Course Teacher:

Course title: Dr.U.P.Kulkarni	Course code: UHUC500
Sem: V	Divison: B
Course Name: Software Engineering& Project Management	

[Academic Year: 2024-25]

Date of submission: -10-2024

Submitted by:

DARSHAN BASAVARJ HALLIGUDI 2SD23CS401

Table of Contents

Contents

1: Write a C program to show that C programming Language support only Call by Value.

```
#include <stdio.h>

void swap(int x,int y){
int temp;
temp=x;
x=y;
y=temp;
printf("inside swap function: x=%d y=%d\n", x ,y);
}

int main(){
int a=10,b=20;
swap(a,b); //swaped by calling function with values in arguments
printf("after calling swap function: a=%d b=%d", a  ,b);
return 0;
}
```

Output:

Inside swap function: x=20 y=10
After calling swap function: a=10 b=20

2: Study the concept "USABILITY". Prepare a report on USABILITY of at least TWO UIs of major software product you have seen.

2.1 Google Drive Usability Analysis

2.2 Overview

Google Drive is a cloud-based file storage and collaboration platform, widely used for personal, educational, and business purposes.

It allows users to store, share, and collaborate on documents, spreadsheets, presentations, and other files, with seamless integration across other Google Workspace applications like Google Docs, Sheets, and Slides.

2.3 Usability Factors

- Google Drive's user interface is simple and intuitive, making it easy for new users to navigate.
- Basic actions like creating, uploading, or sharing files are clearly labeled, with icons that help guide user actions.

- Tutorials and help prompts assist users with advanced features, like setting sharing permissions or using Drive's search capabilities.

Efficiency:

- The search bar at the top is a powerful feature, allowing users to quickly find files using keywords, file types, or collaborators' names, enhancing the speed of organizing and accessing files.
- Users can preview files without needing to download them, which saves time when browsing documents or images.

Error Prevention and Recovery:

- Google Drive features a Trash where deleted files are temporarily stored before being permanently deleted, allowing users to recover files if accidentally deleted.
- Clear prompts before sharing files help prevent unintentional exposure of sensitive information.
- The version history feature allows users to recover previous versions of documents, which is crucial for collaboration.

User Satisfaction:

- Users appreciate Google Drive's simple design, quick access to files, and the ease of sharing and collaborating in real time.
- However, some users find that the organization of files can become cluttered over time, especially when multiple files are shared across various projects.
- The real-time collaboration and ability to integrate with Google Workspace tools are seen as significant advantages.

2.3 Strengths and Weaknesses

Strengths:

- Intuitive interface with simple navigation.
- Powerful search functionality and seamless integration with other Google tools.
- Excellent error recovery through version history and Trash.

Weaknesses:

- File organization can become chaotic with many shared documents.
- Some users may find managing advanced sharing permissions complex.

2.1.1 Microsoft Teams Usability Analysis

2.1.2 Overview

Microsoft Teams is a collaboration tool that integrates chat, video conferencing, and file sharing, making it a popular choice for remote work and communication.

It is part of the Microsoft 365 suite and is used by organizations for internal communication, project management, and team collaboration.

2.1.3 Usability Factors

- Teams has a steeper learning curve compared to Gmail due to its feature-rich environment, including channels, meetings, chats, and file storage.
- Microsoft provides tutorials and in-app guidance to help new users get started, but some users still find the interface overwhelming initially.
- Navigation through different teams, channels, and chats can be confusing for new users.

Efficiency:

- Teams is designed for collaboration, allowing users to share files, schedule meetings, and chat seamlessly.
- Integration with other Microsoft 365 apps (Word, Excel, SharePoint) allows users to edit documents directly within the Teams interface, enhancing productivity.
- The search functionality is robust, but finding specific messages or files in large teams can still be time-consuming.

Memorability:

- Returning users may find it challenging to recall the structure of different channels and teams, especially in large organizations with numerous active conversations.
- The consistent use of icons and color schemes helps users remember how to navigate between chats, teams, and meetings.

Error Prevention and Recovery:

- Teams offers limited undo options for actions like deleting messages or removing files.
- Users can delete their own messages but cannot easily recover them once deleted.

User Satisfaction:

- Users appreciate the comprehensive feature set of Teams, especially its integration with other Microsoft 365 tools.

- The ability to create channels and sub-channels makes it easier for organizations to organize their communication, but it can be overwhelming for new users.
- Some users report a cluttered interface, especially when multiple chats, channels, and calls are active.

2.1.4 Strengths and Weaknesses

Strengths:

- Strong integration with the Microsoft 365 ecosystem.
- Rich feature set for communication and collaboration.
- Allows real-time document collaboration within the interface.

Weaknesses:

- Steeper learning curve, especially for new users.
- User interface can appear cluttered with many active chats and channels.
- Limited options for undoing or recovering deleted messages or files.

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

1. Strong Typing

- ✓ Strong typing means that the language enforces strict data type rules. Variables need to be defined with a specific data type, which prevents errors related to incorrect type usage.
- ✓ Strong typing helps prevent bugs due to unintended type

2.Memory Management

Proper memory management through the use of dynamic memory allocation, deallocation, and avoiding memory leaks ensures that a program runs efficiently.

3. Error Handling

Proper error handling, using mechanisms like conditional checks and error codes, allows programs to anticipate and manage potential errors.

4. Modularity and Functions

- ✓ The ability to break down a program into smaller, reusable functions and modules.
- ✓ Modularity improves code readability, makes testing easier, and allows changes in one part of the code without affecting others.

5.Input validation

```
#include <stdio.h>

int main() {
   int age;
   printf("Enter your age (0-120): ");

   // Check if the input is a valid integer
   if (scanf("%d", &age) != 1) {
      printf("Invalid input! Please enter a number.\n");
      return 1; // Exit if the input is not a number
   }
   // Validate that the age is within the expected range
   if (age < 0 || age > 120) {
      printf("Error: Age must be between 0 and 120.\n");
      return 1; // Exit if the age is out of range
   }
   // If input is valid, print the age
   printf("Your age is: %d\n", age);
      return 0;
}
```

4. 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.

4.1 What is Assertions

An "assertion" is a statement that checks a condition that is expected to be true at a specific point in the code, and if the condition is false, the program usually terminates with an error message.

This mechanism, provided by the assert() macro from the assert.h header file, is crucial for writing reliable code by enabling early detection of invalid states and assumptions within the program.

4.2 Code for Assertion

```
#include <stdio.h>
#include <assert.h>

// Function to calculate the square of a number
int square(int num) {
    // Assert that the input number is non-negative num
    assert(num >= 0);
    return num * num;
}

int main() {
    int number;

    // Prompt user for input
    printf("Enter a non-negative integer: ");
    scanf("%d", &number);

    // Call the square function
    // If the number is negative, the assertion will fail
    int result = square(number);
    printf("The square of %d is: %d\n", number, result);

    return 0;
}
```

Output:
Enter a non-negative integer: 5
The square of 5 is: 25

4.3 **What is POSIX?**

POSIX stands for Portable Operating System Interface, a standard developed by the IEEE (Institute of Electrical and Electronics Engineers) that defines a set of operating system interfaces based on the Unix operating system. It encompasses system APIs, command-line shells, and utility interfaces to ensure a consistent and portable environment across UNIX-like systems.

4.4 POSIX standard using C

```
#include <stdio.h>
```

```
#include <stdlib.h>
#include <unistd.h>   // For fork(), getpid()
#include <sys/wait.h> // For wait()

int main()
{
   pid_t pid = fork(); // Create a child process

   if (pid < 0)
   {
      // Fork failed
      perror("Fork failed");
      exit(EXIT_FAILURE);
   }
   else if (pid == 0)
   {
      // Child process
      printf("Hello from Child! PID: %d\n", getpid());
      exit(0); // Child exits
   }
   else
   {
      // Parent process
      printf("Hello from Parent! PID: %d\n", getpid());

      int status;
      wait(&status); // Wait for the child to finish

      if (WIFEXITED(status))
      {
         printf("Child exited with status: %d\n", WEXITSTATUS(status));
      }
   }

   return 0;
}
```

OUTPUT:

```
Hello from Parent! PID: 17958
Hello from Child! PID: 17959
Child exited with status: 0
```