

**COLLEGE OF COMPUTING AND INFORMATICS SCIENCES**

**SCHOOL OF COMPUTING AND INFORMATION SCIENCES**

**BACHELOR OF SCIENCE IN COMPUTER SCIENCE**

COURSE NAME: BACHELOR OF SCIENCE IN COMPUTER SCIENCE

**A report for assignment 3 (Makerere University Students’ Admission and Placement System) MUSAPS developed by Group 9**

**13th November 2018**

COURSE UNIT: **INDIVIDUALPROJECT 1 ASSIGNMENT FOUR**

COURSE CODE: CSC 1108

GROUP No.: No.9

**GROUP MEMBERS**

|  |  |  |
| --- | --- | --- |
| **NAME** | **REG. No.** | **STUDENT NUMBER** |
| OJOK SIMON PETER | 18/U/25554/EVE | 1800725554 |
| SSERUGO SIMON PETER | 18/U/25565/EVE | 1800725565 |
| KANYONYI ANDREW | 18/U/42097/PS | 1800742097 |

A report submitted in fulfilment of the requirements of Course Unit **Individual Projects 1 CSC 1107**, School of Computing and Informatics Science, CoCIS Makerere University as **assignment four**.

**SUMMERY (ABSTRACT)**

Traditional mode of University Application, Admission, Placement and Registration by use of paper tends to be very tedious, time consuming, filled with irregularities and corruptions and exposes our environment to pollution.

This report aims to document and provide details information on system functionalities and features of Makerere University Students’ Admission and Placement System (MUSAPS) developed by group 9 using C Programming Language. It documents different C components used and their implementations, screenshots of different functionalities as well as the codes and explanations of these screenshots. This reports may also acts as manuals on how the system operates, it can be used as a guide to use and maintenance of the system.

**INTRODUCTION**

Makerere University Students’ Admission and Placement System is a simple and easy to use elementary system which does not require much computer skills developed using C programming Language to aid in Automatic Management of prosperous students’ application, admission, placement, registration and allocation of UACE leavers to different schools, colleges and programs available at Makerere University for 2018-2019 Academic year basing merit and results they obtained in their preceding examination.

Due to limited space, the system allows only and only two students for apply for each of the following combinations done at A level **PCB/M,BCM/IT,PCM/TD,MEG/TD** and **PEM/TD.** Mainly the system has two functionalities, the administrative and students’ functionalities which are accurately explained in the following sections. Those with points below 5 are regarded as non-qualified and ineligible applicants, they are redirected to main menu and quit the system.

**STUDENTS’ SYSTEM FUNCTIONALITIES**

The systems allows new students to create accounts using their desired unique usernames and passwords before they embark on application process as well as old students to login to their already created accounts to proceed with the applications.

The systems uses command prompts to capture students data that a needed during processing and awarding them admissions and placements to different courses, schools, colleges and programs being offered at the university for the particular academic year.

The system has a well-designed users interface with attractive and meaning full green color of Makerere University which can be used by people with little computer skills.

The system does informative functions like displaying application datelines, available courses for particular academic year and help phone numbers and finally on the elapsing of this dateline, it automatically sorts all the applicants and award admission basing on the merit, this processing is done in the background.

The awarding is done on the following procedures, students who got between 20 to 16 points at UACE are given their first choices, from 10 to 15 are given the second choices as long as the course is new and less popular, from 14 to 9 gets their third choices which are non-popular and easy. From 8 to 6 points gets diploma courses if they applied to any and the rest are regarded as non-qualified and ineligible applicants.

Students can view on the command prompt the reported generated for admitted students (Placed Students) and only and only admitted students may proceed to fill and submit their Bio data and non-admitted ones are requested to apply the following academic year. This done by requesting the student to enter his/her names and is search if the names is in the admission report, if found the student proceeds to the Bio Date capture menu. The system also displays the dateline for registration and automatically generates sequentially both students’ and registration numbers in the following format **1800725554** and **18/U/25554** respectively.

On completing filling the Bio Data, the student again re-verified if admitted by asking her/him to re-enter the names and searched in the admitted students’ report if found and filled the Bio Data from, the students proceeds to apply for courses he/she will be doing in the first semester.

**ADMINISTRATIVE SYSTEMS’ FUNCTIONALITIES**

The system allows University Comptrollers, stuff and system Administrator to login to the system using username of **Admin25554** and password **25554** and does the following administrative tasks.

The University comptrollers, stuffs and system Administrators can view lists of students who made choices, Course, schools, colleges and programs they have been admitted to plus time and date of application, points they got at A level, Date of Birth, sex and other personal information.

The system also allow University Comptrollers to view admission lists of all the students inclusive of Course, Schools, Collages and Programs they have been admitted to.

The systems also allows comptrollers to view lists of Registered Students as well those who failed to register before the dateline and their admission are regarded as non-acceptance, at this point students who appealed may be given admission in vacancies left by non-registered ones.

**SCREENSHOOTS OF THE PROGRAM**

**LISTS OF C COMPONENTS USED IN THE PROGRAM**

|  |  |  |  |
| --- | --- | --- | --- |
|  | HEADER FILE | LIBRARY FUNCTION | HOW IT WAS USED |
| 1 | <assert.h> | void assert(int expression) | Add diagnostic to the System Program |
| 2 | <ctype.h> | int isspace(int c) | To check for space in password function |
| 3 | <errno.h> | extern int errno | Declared to detect errors return my functions calls and these errors written to error file |
| 4 | <math.h> | int rand() | Use to generate random numbers for getting students and registration and students numbers |
| 5 | <stdio.h> | int fclose(FILE \*stream) | They are called after writing to a file to close it and clear memory and protect the file content |
| 6 |  | int feof(FILE \*stream) | Used with looping statements as control variable to read data from a given file |
| 7 | FILE \*fopen(const char \*filename, const char \*mode) | Create files for different functions like writing and reading to or from a file |
| 8 | void rewind(FILE \*stream) | Used to rewind file pointer in some functions were a file was used twice for reading and later writing to it |
| 9 | int fprintf(FILE \*stream, const char \*format, ...) | Used to write data stored in a variable to the file stream |
| 10 | int printf(const char \*format, ...) | Sends formatted output to stdout so a user can interact with the program well |
| 11 | int fscanf(FILE \*stream, const char \*format, ...) | To read data from a file and store it to the variable for later use by the program. Eg viewing admission list |
| 12 | int scanf(const char \*format, ...) | To read formatted input from the stdin like keyboard entered by the user for processing ie in during application. |
| 13 | int sscanf(const char \*str, const char \*format, ...) | To read formatted input from a string and these data were used to for verifying the students. |
| 14 | char \*fgets(char \*str, int n, FILE \*stream) | To read a line of texts from the file stream and later used sscanf() to extract some some strings at specific position on the string |
| 15 | int puts(const char \*str) | To write string to the stdout, for example displaying the admission lists |
| 16 | int fgetpos(FILE \*stream, fpos\_t \*pos) | To get file pointer position so that I can write a desired position in the file stream |
| 17 | int fsetpos(FILE \*stream, fpos\_t \*pos) | To set file position to desired location to that we can write data to where we wanted |
| 18 | int fseek(FILE \*stream, long int offset, int whence) | For setting the file position after getting it using the above functions |
| 19 | <stdlib.h> | int atoi(const char \*str) | To Convert the string pointed to, by the argument str to an integer (type int) so I can use it in addition in one of my function |
| 20 |  | int atexit(void (\*func)(void)) | To register my function that is called always when the programming is closing |
| 21 | int system(const char \*string) | With arguments like **clr** and **color 27** I used it to clear the screen and set the console color to green |
| 22 | void srand(unsigned int seed) | To seeds the random number generator to produce un predictable random numbers |
|  | void exit(int status) | To close some running functions to help in memory management |
|  | <string.h> | char \*strcat(char \*dest, const char \*src) | To join two strings to form one, used in getting registration for combing **18/U** with integer **25554** |
|  |  | int strcmp(const char \*str1, const char \*str2) | To compare to strings in giving out schools and colloges, we used this function comparing passwords and usernames |
|  | int strncmp(const char \*str1, const char \*str2, size\_t n) | Used in the similar situation like the one above, to give very accurate result. |
|  | char \*strcpy(char \*dest, const char \*src) | After successful comparison using the above functions I used the function to update data in some of my variables |
|  | char \*strncat(char \*dest, const char \*src, size\_t n) | To accurately compare strings upto the nth charcter, this gave us more accurate result |
|  | size\_t strlen(const char \*str) | Used to determine the size of the strings to determine if the user has entered in the right length information needed for processing |
|  | <time.h> | char \*ctime(const time\_t \*timer) | Get the current time and print it to the file and the console, and determine if the dateline has elapsed |
|  |  | time\_t time(time\_t \*timer) | Getting time for used in the above manner, this function gave time in million seconds since epoch |
|  |  |  |
|  | <conio.h> | Getch() | Get char from the stdin |

**DATA STRUCTURES AND ALGORITHM**

The following data types were used in the program

Primitive data types

1. int
2. char
3. structures(Holding structured before writing them the file)
4. Linked lists(Placing students data before qsorting operations was done)
5. Arrays (used with strings for storing characters)

**Other C Components used**

1. User made preprocessor directives for holding user defined functions and allow the main file to contain few codes for easy maintenance eg interface.h,stud\_functions.h
2. User defined functions
3. File pointer
4. Looping e.g. for, while loops were used to determine program follow, selection like if, switch, if else and if. Else if for determining which code to be executed or skipped

**REFERRENCES**

1. C Standard Library Tutorial 2017 Tutorial Points
2. C How to Program\_ 6th Ed - Paul Deitel and Harvey Deitel
3. Fundamentals of C Programming - Dilum Bandara
4. The C Programming Language (Brian W Kernighan and Den Ritchie)
5. [www.codewithc.com](http://www.codewithc.com)
6. [www.youtube.com](http://www.youtube.com)