UOW MALAYSIA
KDU PENANG
UNIVERSITY COLLEGE

PART OF THE UNIVERSITY OF WOLLONGONG AUSTRALIA GLOBAL NETWORK

ASSIGNMENT COVER PAGE

Programme	**Course Code and Title**
Diploma in Information Technology	DOP1254 Fundamentals of Object-Oriented Programming
Student's name / student's id	**Lecturer's name**
• Lim Zhe Yuan 0204677 • Thor Wen Zheng 0205096 • Tan Peng Heng 0205430	Tan Phit Huan

Date issued	**Submission Deadline**	**Indicative Weighting**
Week 2 – 25/01/2021	Week 6 – 05/03/2021	20%
Assignment 1 title	Musical examination grading system	

This assessment assesses the following course learning outcomes

# as in Course Guide	**UOWM KDU Penang University College Learning Outcome**
CLO1	Express an algorithm using flow chart and pseudo code.

Student's declaration

I certify that the work submitted for this assignment is my own and research sources are fully acknowledged.

Student's signature: LIM ZHE YUAN
THOR WEN ZHENG
TAN PENG HENG

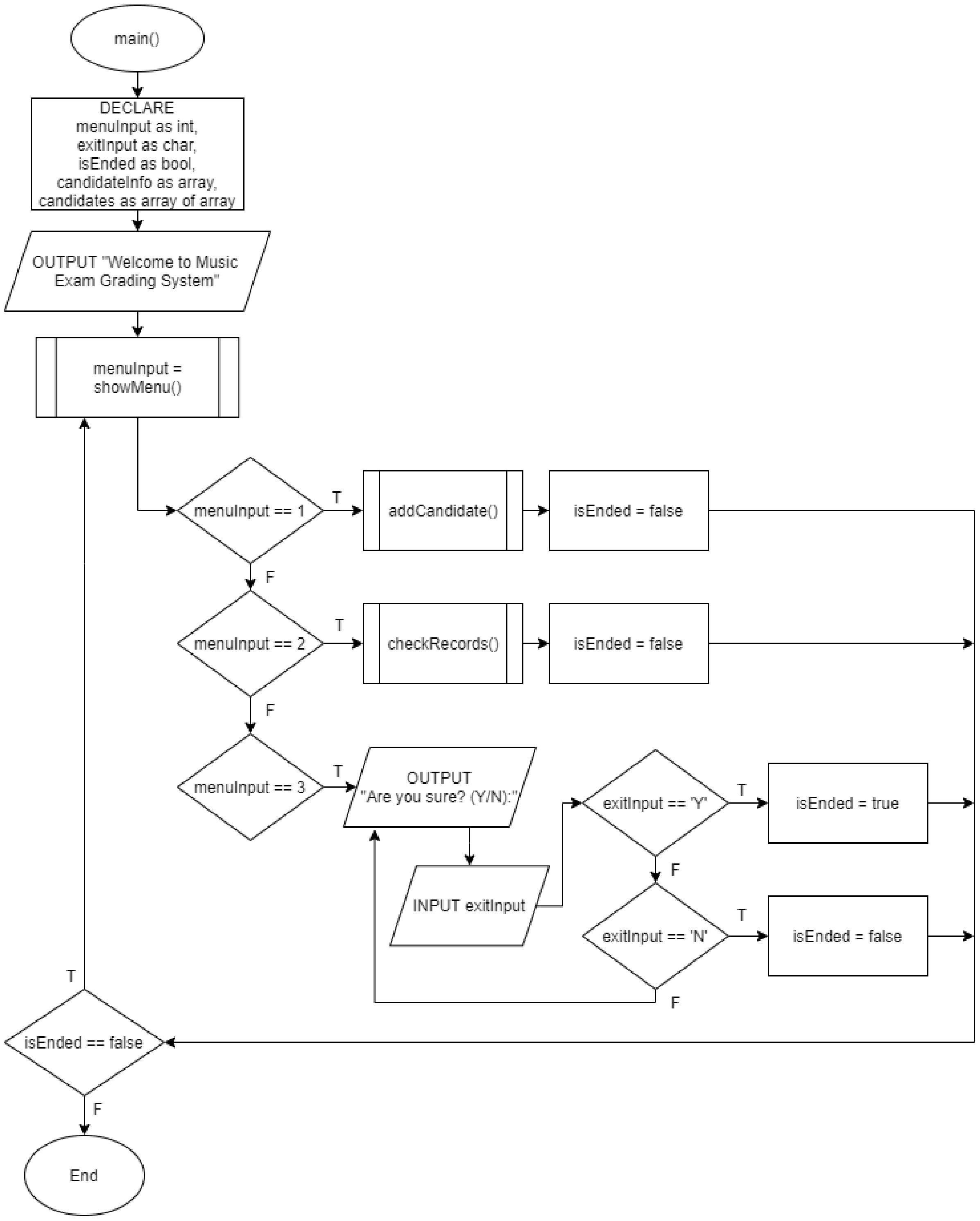
Submission Date: 5 March 2021

Table of Content

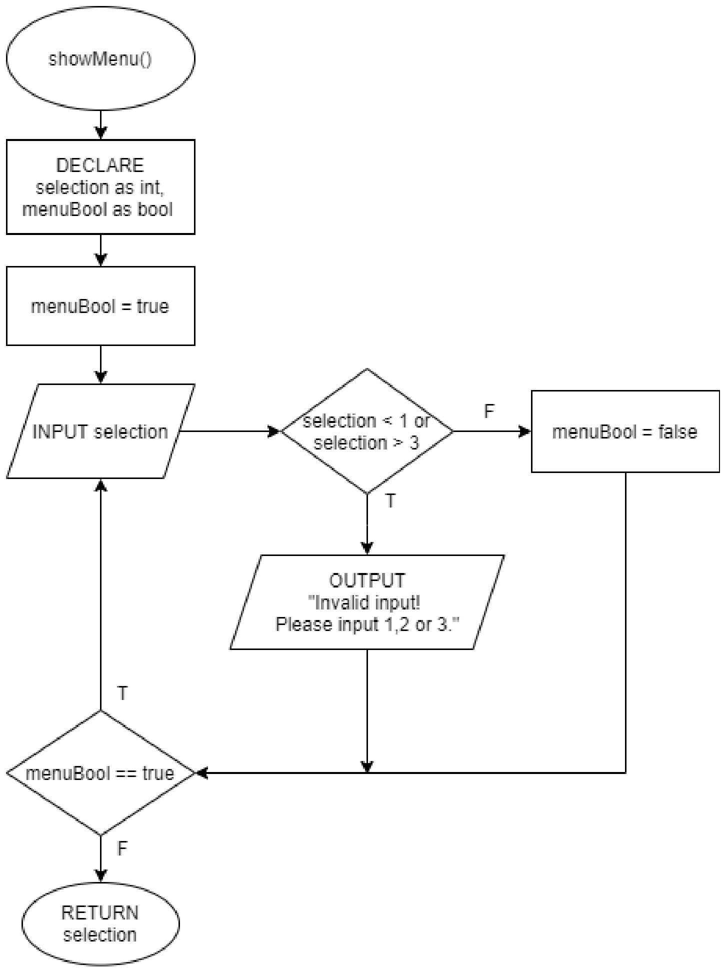
Main Report

Flowcharts

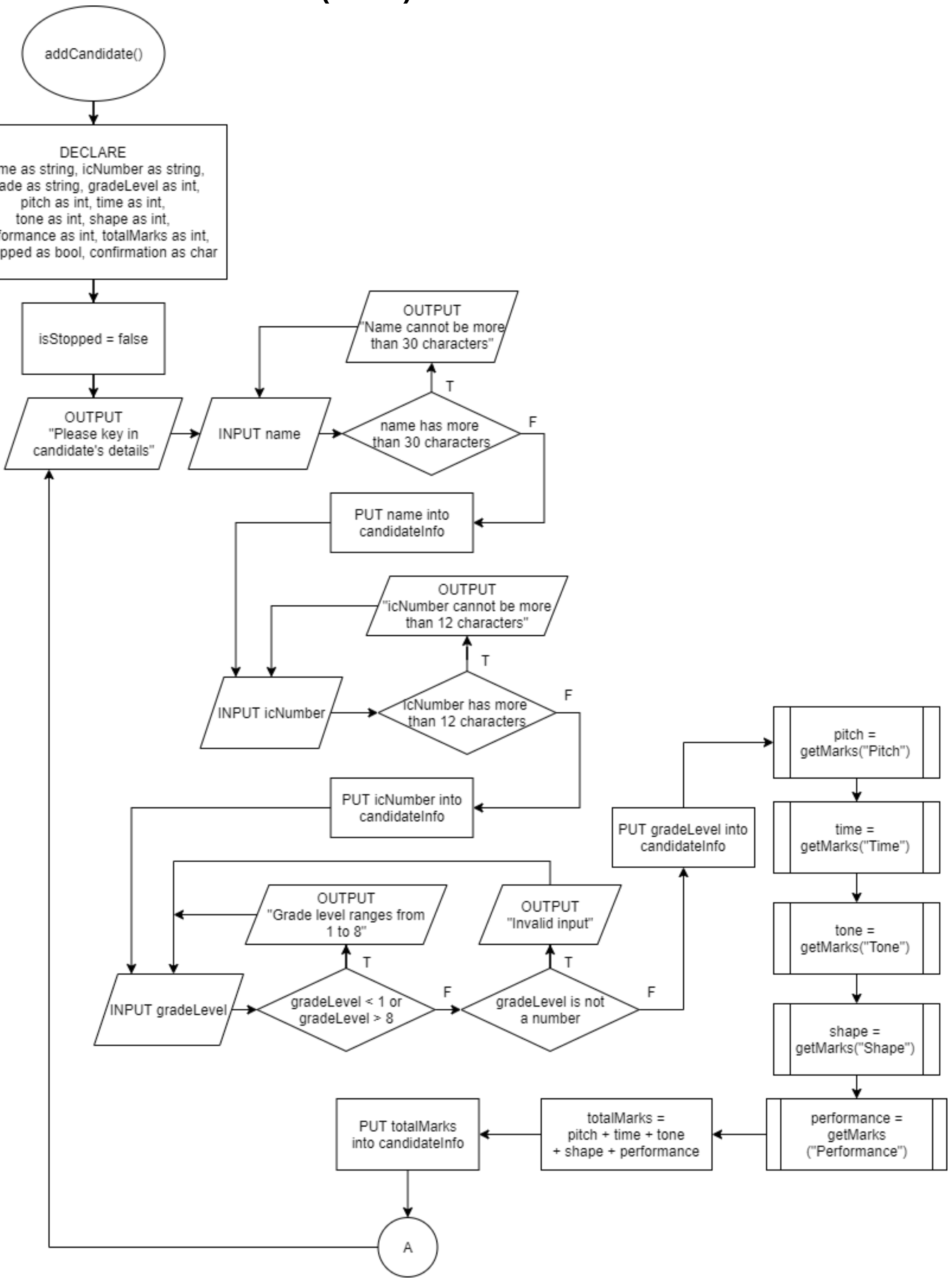
- **Flowchart for main**



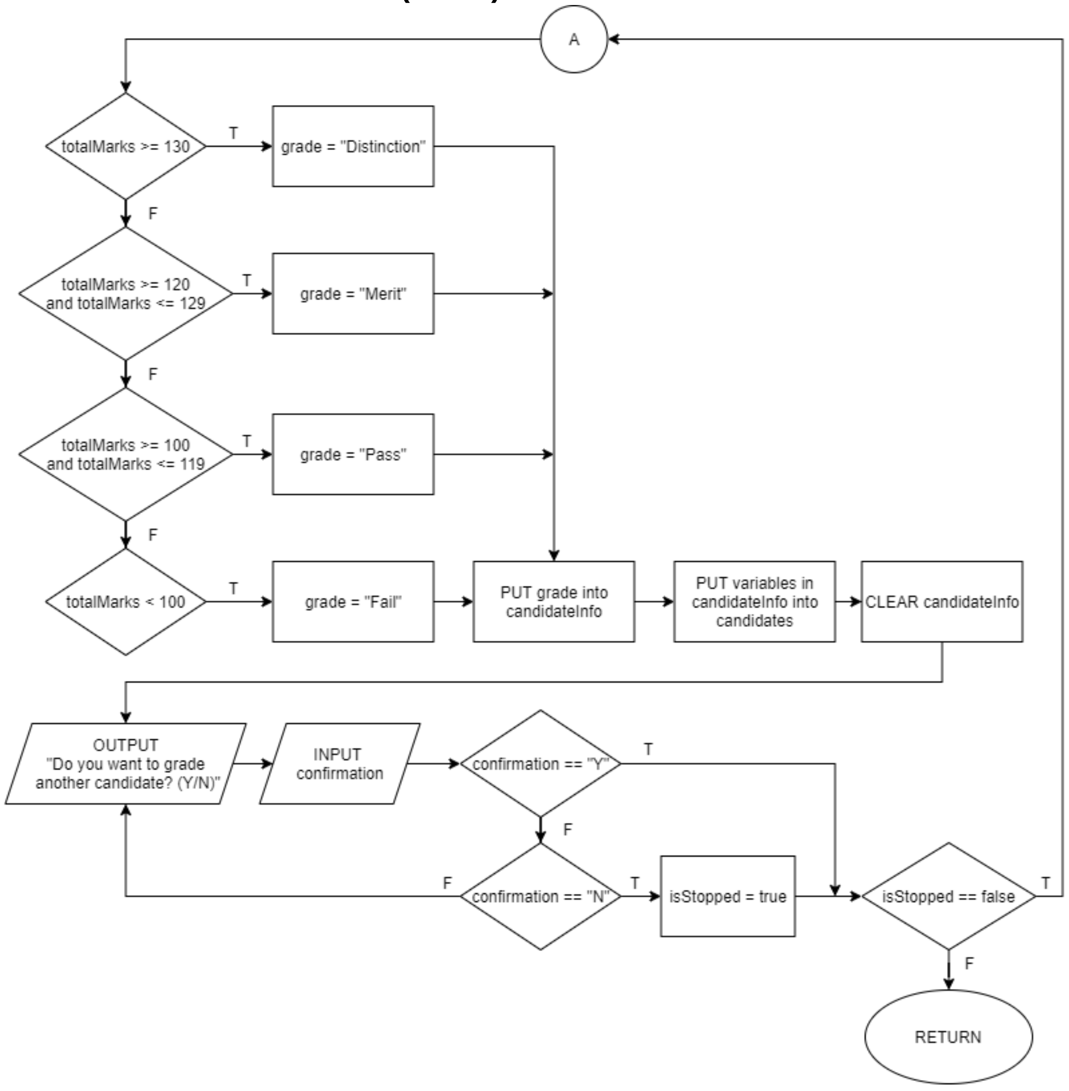
- **Flowchart for showMenu**


showMenu()
DECLARE selection as int, menuBool as bool
menuBool = true
INPUT selection
selection < 1 or selection > 3
F
menuBool = false
T
OUTPUT "Invalid input! Please input 1,2 or 3."
T
menuBool == true
F
RETURN selection

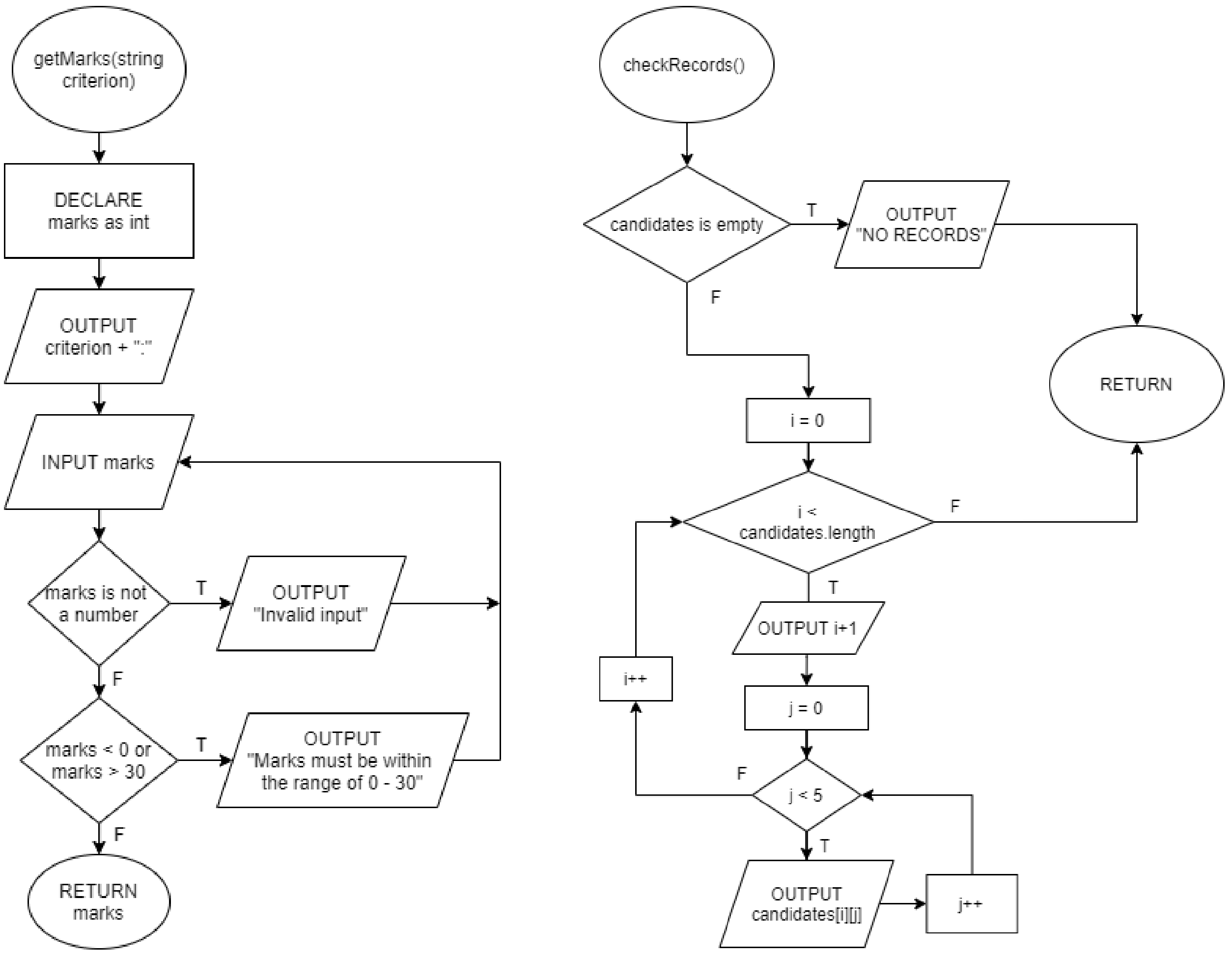
- **Flowchart for addCandidate (Part 1)**


addCandidate()
DECLARE
name as string, icNumber as string,
grade as string, gradeLevel as int,
pitch as int, time as int,
tone as int, shape as int,
performance as int, totalMarks as int,
isStopped as bool, confirmation as char
isStopped = false
OUTPUT "Please key in candidate's details"
INPUT name
name has more than 30 characters
T
OUTPUT "Name cannot be more than 30 characters"
F
PUT name into candidateInfo
INPUT icNumber
icNumber has more than 12 characters
T
OUTPUT "icNumber cannot be more than 12 characters"
F
PUT icNumber into candidateInfo
INPUT gradeLevel
gradeLevel < 1 or gradeLevel > 8
T
OUTPUT "Grade level ranges from 1 to 8"
F
gradeLevel is not a number
T
OUTPUT "Invalid input"
F
PUT gradeLevel into candidateInfo
pitch = getMarks("Pitch")
time = getMarks("Time")
tone = getMarks("Tone")
shape = getMarks("Shape")
performance = getMarks ("Performance")
totalMarks = pitch + time + tone + shape + performance
PUT totalMarks into candidateInfo
A

- **Flowchart for addCandidate (Part 2)**



- **Flowchart for getMarks and checkRecords**



Complete Program

/* NOTE: In Dev-C++, the error "[Error] to_string was not declared in this scope" may occur when compiling program

SOLUTION: Go to Tools > Compiler Options > Settings > Code Generation > Language standard (-std): set to "ISO C++11" */

```
#include <iostream>
#include <iomanip>          // For input handling and menu output customization
#include <string>           // For using string functions
#include <cctype>           // For toupper()
#include <limits>           // For numeric_limits<streamsize>::max() in cin.ignore()
#include <vector>           // For using vectors - candidates and candidateInfo
using namespace std;

// Function declarations
int showMenu();
void addCandidate();
void checkRecords();
int getMarks(string criterion);

// Vector initialization
vector< vector<string> > candidates;
vector<string> candidateInfo;

int main() {
        // Declare menu variables
        int menuInput;
        char exitInput;
        bool isEnded = false;
```

```
	// Print welcome message
	cout << "Welcome to Practical Music Examination Grading System\n\n";

	// Flag-controlled do...while loop, ends if isEnded == true
	do {
		// Show the menu and assign selection value to menuInput
		menuInput = showMenu();

		// Redirect user to selected function
		if (menuInput == 1) {
			addCandidate();
			isEnded = false;
		} else if (menuInput == 2) {
			checkRecords();
			isEnded = false;
		} else if (menuInput == 3) {
			// Exit
			do {
				cout << "Are you sure? (Y/N): ";
				cin >> exitInput;
				// Flush input stream incase user types more than one char,
				// which may result in the extra input going into the next cin ->
SELECTION input

				cin.ignore(numeric_limits<streamsize>::max(), '\n');
				exitInput = toupper(exitInput);

				if (exitInput == 'Y') {
					isEnded = true;
				} else if (exitInput == 'N') {
					isEnded = false;
```

```
                    }
                } while (exitInput != 'Y' && exitInput != 'N');
            }
        } while (isEnded == false);

        return 0;
}

int showMenu() {
        // Declare variables
        int selection;
        bool menuBool = true;

        do {
                // Display the menu
                cout.fill('_');
                cout << endl << setw(46) << right << "_\n";
                cout.fill(' ');
                cout << setw(23) << right << "Menu\n";
                cout.fill('_');
                cout << setw(46) << right << "_\n";
                cout << "\n          " << "1. Add candidate\n"
                     << "          " << "2. Check records\n"
                     << "          " << "3. Exit\n";

                // Handle user input
                cout << "\nSelection: ";
                cin >> selection;
                // If user input does not match datatype of variable, cin will enter fail state
(do...while loops infinitely)
```

```
			// this if block below clears fail state and flushes "bad input" from input stream
			if (cin.fail()) {
				cin.clear();
				cin.ignore(numeric_limits<streamsize>::max(), '\n');
			} else {
				// Flush input stream regardless of fail state
				cin.ignore(numeric_limits<streamsize>::max(), '\n');
			}

			// Check input
			if (selection < 1 || selection > 3) {
				cout << "\n* * * Invalid input. Please input 1, 2, or 3. * * *\n";
				menuBool = true;
			} else {
				menuBool = false;
			}

		} while (menuBool == true);

		return selection;
}

void addCandidate() {
		// Declare variables to store candidate info and marks for criteria
		string name, icNumber, grade;
		int gradeLevel, pitch, time, tone, shape, performance, totalMarks;

		// Variables used for the condition of outer do...while loop
		bool isStopped = false;
		char confirmation;
```

```
	do {
		// Prompt for and record candidate details into temporary vector
		cout << "\n\nPlease key in the candidate's details.\n";
		cout.fill('_');
		cout << setw(84) << right << "_\n";

		// Prompt for name
		do {
			cout << "Name: ";
			getline(cin, name);
			if (name.length() > 30)
				cout << "* * * Name cannot be longer than 30 characters * * *\n";
		} while (name.length() > 30);
		candidateInfo.push_back(name);

		// Prompt for I/C Number
		do {
			cout << "Identity Card Number: ";
			getline(cin, icNumber);
			if (icNumber.length() > 12)
				cout << "* * * I/C Number cannot be more than 12 characters * *
*\n";
		} while (icNumber.length() > 12);
		candidateInfo.push_back(icNumber);

		// Prompt for Grade Level
		do {
			cout << "Grade level: ";
			cin >> gradeLevel;
```

```
        if (cin.fail()) {
            cout << "* * * Invalid input * * *\n";
            cin.clear();
            cin.ignore(numeric_limits<streamsize>::max(), '\n');
        } else {
            if (gradeLevel < 1 || gradeLevel > 8)
                cout << "* * * Grade level ranges from 1 to 8 * * *\n";
            cin.ignore(numeric_limits<streamsize>::max(), '\n');
        }
    } while (gradeLevel < 1 || gradeLevel > 8);
    candidateInfo.push_back(to_string(gradeLevel));

    // Prompt for candidate's marks for each criterion
    cout << "\nPlease key in the marks for each marking criterion. (0 - 30 marks per
criterion)\n";
    cout.fill('_');
    cout << setw(84) << right << "_\n";

    pitch = getMarks("Pitch");
    time = getMarks("Time");
    tone = getMarks("Tone");
    shape = getMarks("Shape");
    performance = getMarks("Performance");

    // Calculate total marks and store in temporary vector
    totalMarks = pitch + time + tone + shape + performance;
    candidateInfo.push_back(to_string(totalMarks));

    // Determine grade and store in temp vector
    if (totalMarks >= 130) {
            grade = "Distinction";
    } else if (totalMarks >= 120 && totalMarks <= 129) {
            grade = "Merit";
    } else if (totalMarks >= 100 && totalMarks <= 119) {
            grade = "Pass";
    } else if (totalMarks < 100) {
            grade = "Fail";
    }
    candidateInfo.push_back(grade);

    // Display final marks and grade
    cout.fill('_');
    cout << setw(84) << right << "_";
    cout << "\n| Total marks: " << totalMarks
             << " | Grade: " << grade << " |\n\n";

    // Add this candidate's info, marks, and grade (in temp vector) into 2D vector
"candidates"
    // Then clear the temporary vector used for this current candidate
    candidates.push_back(candidateInfo);
    candidateInfo.clear();

    // Ask user if they want to continue or return to menu
    do {
            cout << "Do you want to grade another candidate? (Y/N): ";
            cin >> confirmation;
            cin.ignore(numeric_limits<streamsize>::max(), '\n');

            confirmation = toupper(confirmation);
            if (confirmation == 'Y') {
                isStopped = false;
            } else if (confirmation == 'N') {
                isStopped = true;
            }
        } while (confirmation != 'Y' && confirmation != 'N');

    } while (isStopped == false);
}

void checkRecords() {
    // Check if candidates vector is empty
    if (candidates.empty()) {
        cout << endl;
        cout.fill('=');
        cout << setw(116) << right << "=\n";
        cout.fill(' ');
        cout << setw(63) << right << "NO RECORDS";
        cout.fill('=');
        cout << setw(116) << left << "\n=";
        cout << endl;
    } else {
        // Print table if candidates vector is not empty
        // Print table header
        cout << endl;
        cout.fill('=');
        cout << setw(116) << right << "=\n";
        cout.fill(' ');
        cout << setw(6) << left << "No.";
        cout << setw(34) << left << "Name";
        cout << setw(19) << left << "I/C Number";
```

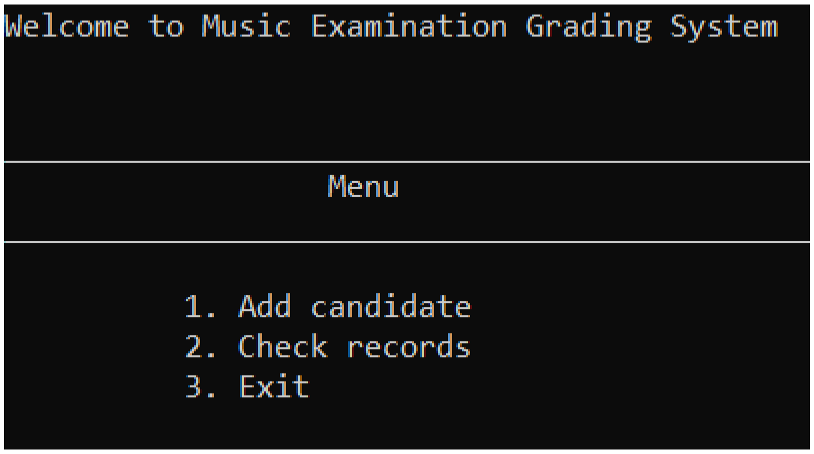
```
                cout << setw(19) << left << "Grade Level";
                cout << setw(19) << left << "Total Marks";
                cout << setw(19) << "Grade";
                cout.fill('=');
                cout << setw(116) << left << "\n=";
                cout << endl;

                // Print table contents
                cout.fill(' ');
                for (int i = 0; i < candidates.size(); i++) {
                        cout << i + 1 << ".    ";

                        for (int j = 0; j < candidates[i].size(); j++) {
                                // Larger space for Name column, smaller space for subsequent
columns
                                if (j == 0) {
                                        cout << setw(34) << left << candidates[i][j];
                                } else {
                                        cout << setw(19) << left << candidates[i][j];
                                }
                        }
                        cout << endl;
                }
        }
}

int getMarks(string criterion) {
        int marks;

        // Prompt user for marks
```

```
    do {
        cout << criterion << ": ";
        cin >> marks;
        // Catch invalid inputs that may cause cin fail state
        // Then check if marks is valid
        if (cin.fail()) {
            cout << "* * * Invalid input * * *\n";
            marks = 69;
            // Clear fail state
            cin.clear();
            cin.ignore(numeric_limits<streamsize>::max(), '\n');
        } else {
            if (marks < 0 || marks > 30) {
                cout << "* * * Marks must be within "
                     << "the range of 0 - 30 * * *\n";
            }
            // Flush input stream to get rid of any excess input
            cin.ignore(numeric_limits<streamsize>::max(), '\n');
        }
    } while (marks < 0 || marks > 30);

    return marks;
}
```

Description of the Program

- ### Introduction & Main Menu

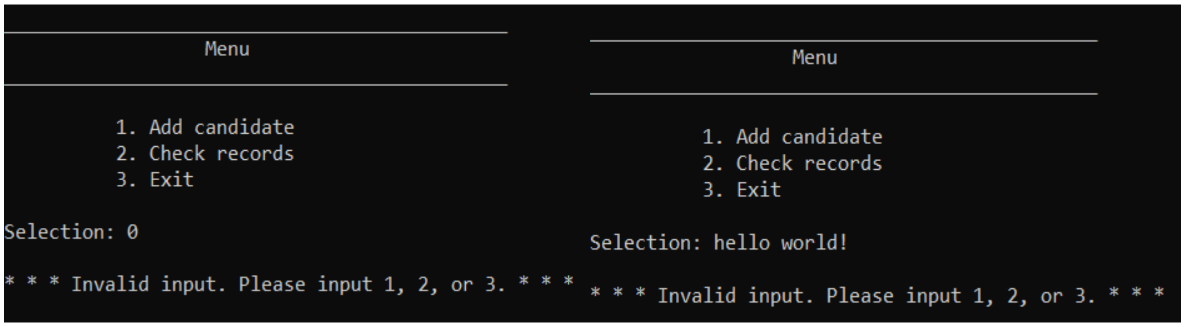
This system is a grading system for the examiner of a practical graded music exam to handle the results of exam candidates. When the system starts, a welcome message is displayed to the user and then the main menu of the system is also displayed. The main menu shows 3 options, the first option is “Add candidate”, the second option is “Check records”, and the third option is “Exit”. These 3 options are basically the main functions of the system.



```
Welcome to Music Examination Grading System

________________________________________
                 Menu
________________________________________

         1. Add candidate
         2. Check records
         3. Exit
```

Figure 1.0: Welcome message and main menu of the system.

Immediately after the main menu is shown, the user will be prompted for a “Selection” input, which represents which function in the system the user selects. For the “Selection” input, the system only accepts the values 1, 2, or 3; if the user inputs any other value, the system will notify the user that the input is invalid and remind the user what the acceptable inputs are, then it redisplays the menu and prompts the user for another input. The system will keep prompting the user for a valid input until the user inputs any value from 1, 2, or 3. Upon receiving a valid input, the system redirects the user to the specific system function that they selected.

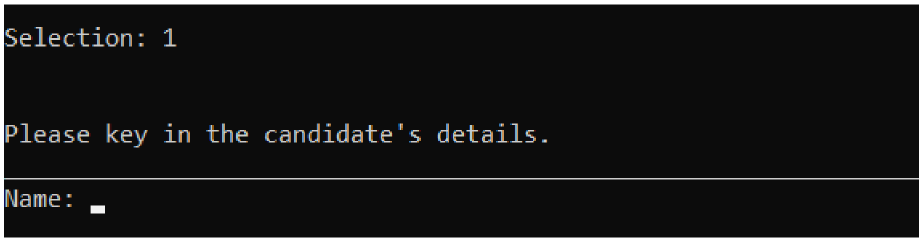


```
________________________________________
                 Menu
________________________________________

         1. Add candidate
         2. Check records
         3. Exit

Selection: 0

* * * Invalid input. Please input 1, 2, or 3. * * *
```

```
________________________________________
                 Menu
________________________________________

         1. Add candidate
         2. Check records
         3. Exit

Selection: hello world!

* * * Invalid input. Please input 1, 2, or 3. * * *
```

Figure 1.1: Invalid inputs are rejected; the system continuously prompts the user for valid input.

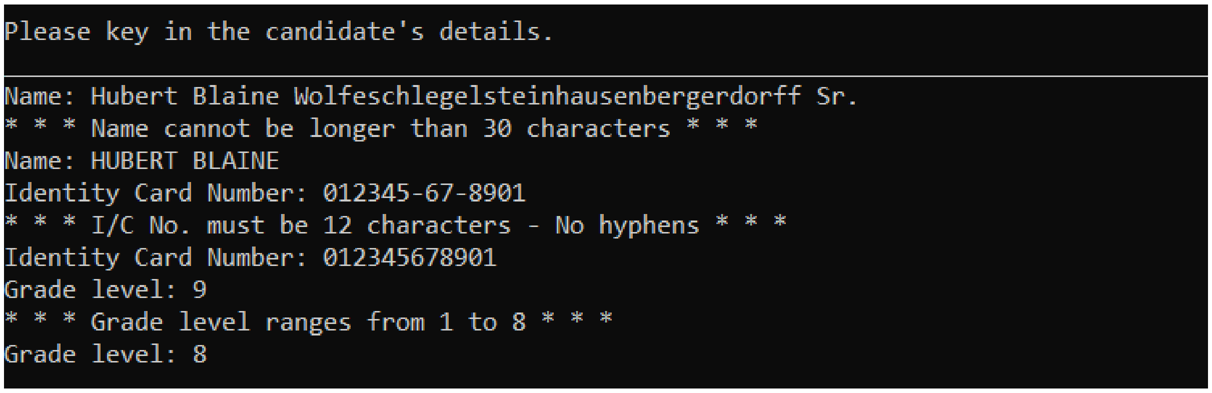
- **Add Candidate**



```
Selection: 1


Please key in the candidate's details.
Name: 
```

Figure 2.0: User inputs 1, system redirects user to Add Candidate function.

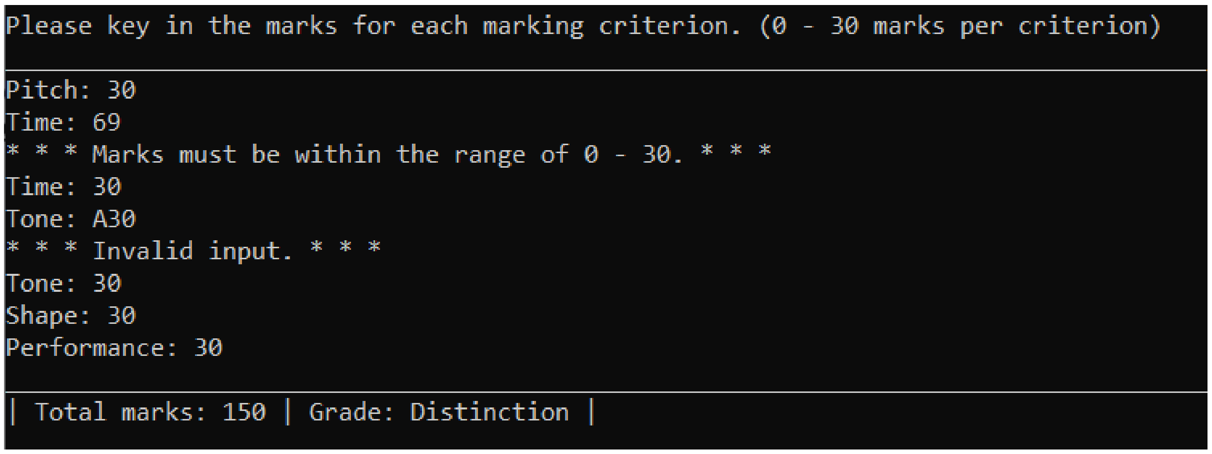
In the case that the user inputs "1" in the menu, the system will redirect the user to the "Add candidate" function. The purpose of the "Add candidate" function is to allow the user to grade and add a new exam candidate into the system. In this function, the system first prompts the user to enter the details of the candidate, including the candidate's name, identity card number, and grade level. Names cannot be longer than 30 characters, identity card numbers cannot be longer than 12 characters, while grade level must be a number from 1 to 8.



```
Please key in the candidate's details.
Name: Hubert Blaine Wolfeschlegelsteinhausenbergerdorff Sr.
* * * Name cannot be longer than 30 characters * * *
Name: HUBERT BLAINE
Identity Card Number: 012345-67-8901
* * * I/C No. must be 12 characters - No hyphens * * *
Identity Card Number: 012345678901
Grade level: 9
* * * Grade level ranges from 1 to 8 * * *
Grade level: 8
```

Figure 2.1: Process of entering candidate details.

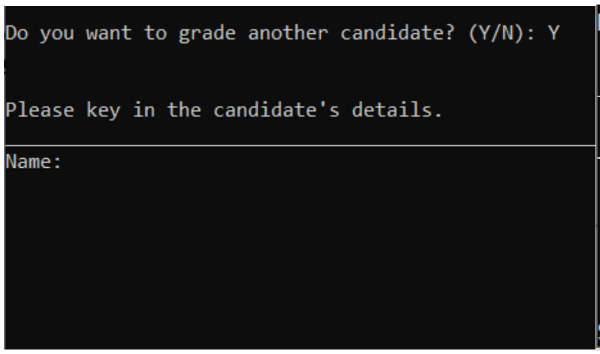
Next, the system prompts the user to key in the marks for each marking criterion in the music exam. There is a total of 5 marking criteria, namely pitch, time, tone, shape, and performance. For each criterion, the system only accepts numbers between 0 to 30; it rejects the input if the marks are outside the range of 0 to 30, or if the input starts with a character. If the input is invalid, the user should refer to the relevant error messages to figure out what went wrong. The system will prompt the user to re-enter the input for the current marking criterion until the user inputs an acceptable value. Once the marks for all the marking criteria have been keyed in, the system calculates the total marks, determines the final grade based on the total mark, and then displays the total marks and final grade to the user.



```
Please key in the marks for each marking criterion. (0 - 30 marks per criterion)

Pitch: 30
Time: 69
* * * Marks must be within the range of 0 - 30. * * *
Time: 30
Tone: A30
* * * Invalid input. * * *
Tone: 30
Shape: 30
Performance: 30

| Total marks: 150 | Grade: Distinction |
```

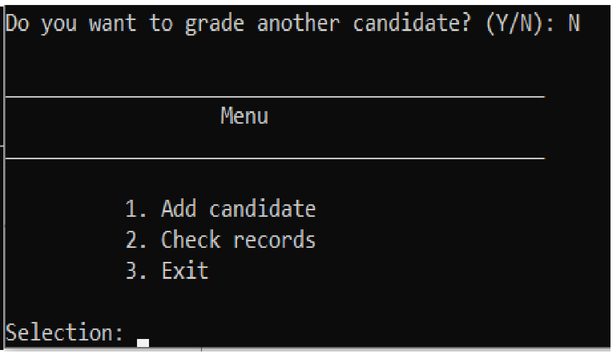
Figure 2.2: Process of handling results of candidates

Then, the system asks the user if they want to grade another candidate. If the user enters 'Y', the system redirects the user to the beginning of the "Add candidate" function, and the user will be prompted to enter candidate details and marks once again. Conversely, if the user enters 'N', the "Add candidate" function ends and the system redirects the user to the main menu of the system, where the user is prompted for a "Selection" input again.



```
Do you want to grade another candidate? (Y/N): Y


Please key in the candidate's details.

Name:
```

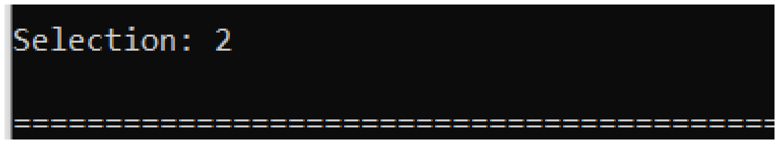
Figure 2.3: User inputs 'Y'



```
Do you want to grade another candidate? (Y/N): N

                    Menu

          1. Add candidate
          2. Check records
          3. Exit

Selection:
```

Figure 2.4: User inputs 'N'

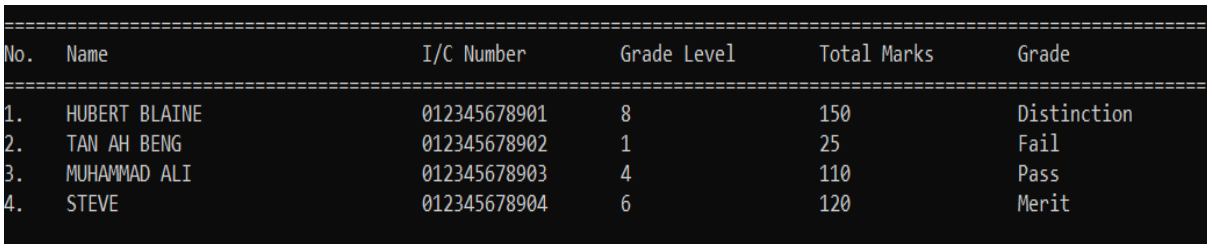
- **Check Records**



```
Selection: 2

==============================================
```

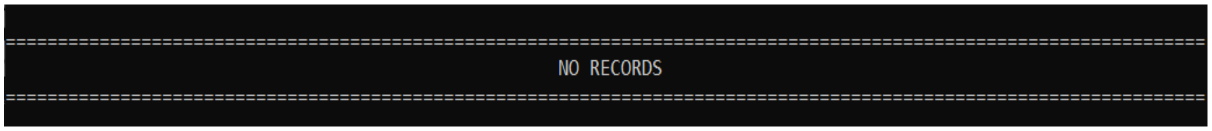
Figure 3.0: User inputs 2, system redirects user to “Check records” function

If the user inputs “2” in the menu, the system redirects the user to the “Check records” function. The purpose of the “Check records” function is to allow the user to check the details, total marks, and final grade of all previously recorded exam candidates. In this function, the system will display the details, total marks, and final grade of all recorded exam candidates in a table. If the user had not recorded any exam candidates in the system yet, the system simply shows “NO RECORDS” to the user.



```
==========================================================================================================
No.   Name                             I/C Number       Grade Level       Total Marks       Grade
==========================================================================================================
1.    HUBERT BLAINE                    012345678901     8                 150               Distinction
2.    TAN AH BENG                      012345678902     1                 25                Fail
3.    MUHAMMAD ALI                     012345678903     4                 110               Pass
4.    STEVE                            012345678904     6                 120               Merit
```

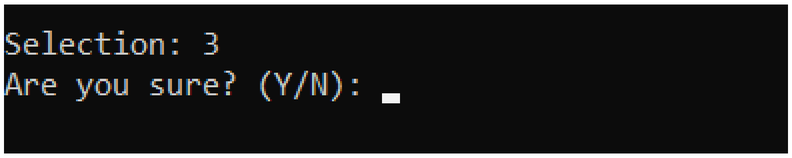
Figure 3.1: System shows a table containing all previous records



```
==========================================================================================================
                                              NO RECORDS
==========================================================================================================
```

Figure 3.2: System shows “NO RECORDS” if no candidate has been added yet

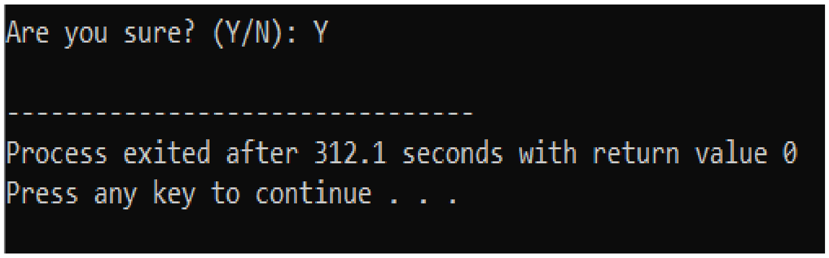
- **Exit**



```
Selection: 3
Are you sure? (Y/N): _
```

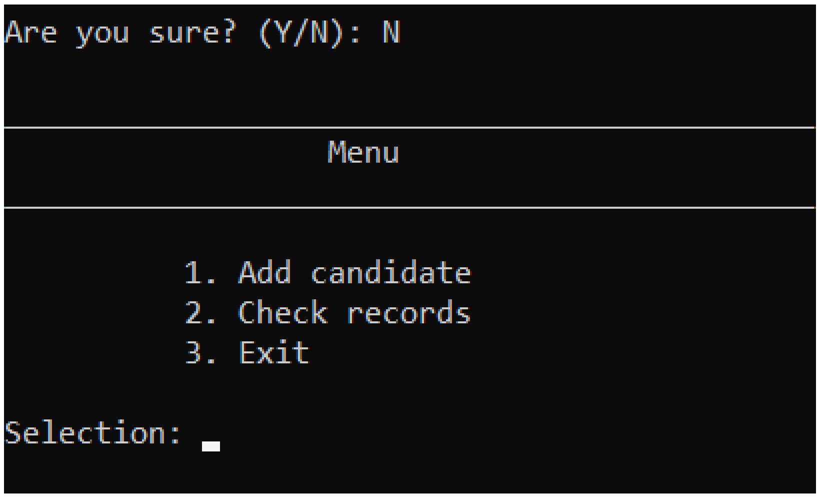
Figure 4.0: User inputs 3, system redirects user to “Exit” function

Lastly, if the user inputs “3” in the menu, the system redirects the user to the “Exit” function. This function can be used by the user to end and quit the system. As seen in ***Figure 4.0***, the system displays a message and prompts the user for confirmation to quit the system or not. If the user inputs ‘Y’, the system ends; if the user inputs ‘N’, the system does not end, and it redirects the user back to the main menu.



```
Are you sure? (Y/N): Y

--------------------------------
Process exited after 312.1 seconds with return value 0
Press any key to continue . . .
```

Figure 4.1: User inputs ‘Y’; system ends



```
Are you sure? (Y/N): N

                    Menu

          1. Add candidate
          2. Check records
          3. Exit

Selection: _
```

Figure 4.2: User inputs ‘N’; user is redirected to main menu

Bibliography List

cplusplus.com (2020) *Vector.* cplusplus.com. Available from http://cplusplus.com/reference/vector/vector/ [accessed 6 February 2021]

cplusplus.com (2020) *type_info::name*. cplusplus.com. Available from http://www.cplusplus.com/reference/typeinfo/type_info/name/ [accessed 6 February 2021]

cplusplus.com (2020) *to_string*. cplusplus.com. Available from http://www.cplusplus.com/reference/string/to_string/ [accessed 6 February 2021]

cplusplus.com (2020) <*iomanip*>. cplusplus.com. Available from https://www.cplusplus.com/reference/iomanip/ [accessed 6 February 2021]

cplusplus.com(2020) *toupper.* cplusplus.com. Available from http://www.cplusplus.com/reference/cctype/toupper/ [accessed 6 February 2021]

Malik, D.S. (2008) *Introduction to C++ Programming: Brief Edition*. Massachusetts, USA: Cengage Course Technology.

Pai, A. (2014) *Validating user input in C++*. hackerearth. Available from https://www.hackerearth.com/practice/notes/validating-user-input-in-c/ [accessed 6 February 2021]