

Figure

Foundation Certificate for Higher Education

Module: DOC 333 Introduction to Programming Principles

Module Leader: Mr. Sudarshan Welihinda

Assignment Type: Individual

Submission Date: 2023.12.11

Student ID: 20231264

Student Name: Ranuga Disansa Belpa Gamage

Student Email: [ranuga.20231264@iit.ac.lk](mailto:ranuga.20231264@iit.ac.lk)

Acknowledgement

I want to express my appreciation to those who have helped complete this report.

I have been able to complete this report because of the Academic advisor Mr. Sudarshan Welihinda for their invaluable support, mentorship, and feedback and the faculty members of Doc 333 (Introduction to Programming Principles).

Additionally, I would like to thank my family members for their unwilling encouragement and help throughout the report.

Thank you.

Table of Contents

[Introduction 5](#_Toc152980156)

[The Problem 5](#_Toc152980157)

[The Solution 5](#_Toc152980158)

[Algorithm 6](#_Toc152980159)

[PesudoCode 6](#_Toc152980160)

[Verification Functions 11](#_Toc152980161)

[date\_vertification() 11](#_Toc152980162)

[project\_code\_vertification() 13](#_Toc152980163)

[check\_if\_int() 13](#_Toc152980164)

[Explanation 14](#_Toc152980165)

[Add a new project to existing projects. 14](#_Toc152980166)

[Removing a Completed project from existing projects 14](#_Toc152980167)

[Adding new workers to the available worker's group 15](#_Toc152980168)

[Updating Details to an ongoing project 15](#_Toc152980169)

[Project Statistics 16](#_Toc152980170)

[Exiting the Program 16](#_Toc152980171)

[Assumptions 17](#_Toc152980172)

[Python Code 18](#_Toc152980173)

[Test Cases 22](#_Toc152980174)

[Add a New Project 22](#_Toc152980175)

[Test 1 22](#_Toc152980176)

[Test 2 22](#_Toc152980177)

[Test 3 23](#_Toc152980178)

[Test 4 23](#_Toc152980179)

[Test 5 24](#_Toc152980180)

[Test 6 25](#_Toc152980181)

[Remove a Project 26](#_Toc152980182)

[Assuming the project below has been added 26](#_Toc152980183)

[Test 1 26](#_Toc152980184)

[Test 2 27](#_Toc152980185)

[Test 3 27](#_Toc152980186)

[Add new Workers 28](#_Toc152980187)

[Test 1 28](#_Toc152980188)

[Test 2 28](#_Toc152980189)

[Test 3 29](#_Toc152980190)

[Test 4 29](#_Toc152980191)

[Update Project Details 30](#_Toc152980192)

[Assuming the project below has been added 30](#_Toc152980193)

[Test 1 30](#_Toc152980194)

[Test 2 31](#_Toc152980195)

[Test 3 31](#_Toc152980196)

[Test 4 32](#_Toc152980197)

[Test 5 33](#_Toc152980198)

[Test 6 33](#_Toc152980199)

[Test 7 34](#_Toc152980200)

[Test 8 35](#_Toc152980201)

[Project Statistics 36](#_Toc152980202)

[Assuming the project below has been added from Choice 3 36](#_Toc152980203)

[Assuming the project below has been added from Choice 1 36](#_Toc152980204)

[Test 1 36](#_Toc152980205)

[Test 2 37](#_Toc152980206)

[Exit 38](#_Toc152980207)

[Test 1 38](#_Toc152980208)

List of Figures

[Figure 1 0](file:///D:\University\IIT\DOC333\DOC333-CW-Sem1\Report\20231264_final.docx#_Toc152980242)

[Figure 2 11](#_Toc152980243)

[Figure 3 12](#_Toc152980244)

[Figure 4 13](#_Toc152980245)

[Figure 5 13](#_Toc152980246)

[Figure 6 14](#_Toc152980247)

[Figure 7 14](#_Toc152980248)

[Figure 8 15](#_Toc152980249)

[Figure 9 15](#_Toc152980250)

[Figure 10 16](#_Toc152980251)

[Figure 11 18](#_Toc152980252)

[Figure 12 18](#_Toc152980253)

[Figure 13 19](#_Toc152980254)

[Figure 14 19](#_Toc152980255)

[Figure 15 20](#_Toc152980256)

[Figure 16 21](#_Toc152980257)

[Figure 17 21](#_Toc152980258)

[Figure 18 21](#_Toc152980259)

[Figure 19 22](#_Toc152980260)

[Figure 20 22](#_Toc152980261)

[Figure 21 23](#_Toc152980262)

[Figure 22 24](#_Toc152980263)

[Figure 23 25](#_Toc152980264)

[Figure 24 26](#_Toc152980265)

[Figure 25 27](#_Toc152980266)

[Figure 26 27](#_Toc152980267)

[Figure 27 28](#_Toc152980268)

[Figure 28 28](#_Toc152980269)

[Figure 29 29](#_Toc152980270)

[Figure 30 29](#_Toc152980271)

[Figure 31 30](#_Toc152980272)

[Figure 32 30](#_Toc152980273)

[Figure 33 31](#_Toc152980274)

[Figure 34 32](#_Toc152980275)

[Figure 35 32](#_Toc152980276)

[Figure 36 33](#_Toc152980277)

[Figure 37 34](#_Toc152980278)

[Figure 38 35](#_Toc152980279)

[Figure 39 35](#_Toc152980280)

[Figure 40 37](#_Toc152980281)

[Figure 41 37](#_Toc152980282)

[Figure 42 38](#_Toc152980283)

# Introduction

## The Problem

The company “XYZ” a company that undertakes large housing construction projects needs an information system to maintain details of the projects they undertake. This system should keep details of all ongoing projects, the details and the number of workers available to assign to a new project. Before undertaking a project, the company makes sure they have enough workers if not the company doesn’t undertake the project. Once the project is finished it is taken out of the ongoing projects and assigned to completed projects and the workers are released when the project is completed.

## The Solution

The solution that is created is for the problem to maintain the details of projects the construction company named “XYZ” undertakes. The solution is an information system built using Python programming language.

# Algorithm

The solution which is implemented by Python Programming language is stated below in the form of algorithm steps and with an explanation of how each aspect of the program functions.

## PesudoCode

1. Start
2. # importing packages

* IMPORT datetime

1. # initialization of variables

* SET company\_name TO “XYZ Company.”
* SET workers TO 0
* SET choice TO 0
* SET all\_projects TO []
* SET completed\_projects TO []
* SET execute TO True
* SET project\_names TO []
* SET possible\_inputs TO [“ongoing”, “completed”, “on hold”]
* SET statistics\_list TO [0] \* len(possible\_inputs)
* SET redirect\_choice TO False
* SET redirect\_to TO None

1. define function menu(redirect,to,company\_name,msg):

* SET main\_menu TO company\_name + ”””

Main Menu

1. Add a new project to existing projects.
2. Remove a completed project from existing projects
3. Add new workers to available workers group
4. Update details on ongoing projects
5. Project Statistics
6. Exit

“””

* OUTPUT “Redirecting…” if redirect is True else main\_menu
* RETURN to if redirect is True else INPUT user choice

1. define function remove\_completed\_projects(code\_of\_project,every\_project,workers\_tot,stats\_list,complete\_projects,possible\_stats):

* TRY
* SET index\_of\_project TO project\_names.index(code\_of\_project)
* SET date\_time TO datetime.datetime.now()
* SET actual\_end\_date TO date\_time.strftime(“%m/%d/%Y”)
* SET code\_of\_project,clients\_name,start\_date,expected\_end\_date,number\_of\_workers,\_,index TO every\_project[index\_of\_project]
* SET completed\_project\_details TO [code\_of\_project,clients\_name,start\_date,expected\_end\_date,number\_of\_workers,actual\_end\_date]
* SET workers\_tot TO workers\_tot + num\_of\_workers
* SET stats\_list[index] TO stats\_list[index] - 1
* SET stats\_list[possible\_stats.index(“completed”)] TO stats\_list[possible\_stats.index(“completed”)] + 1
* APPEND completed\_project\_details TO completed\_projects
* DELETE every\_project[index\_of\_project]
* DELETE project\_names[index\_of\_project]
* RETURN (True, “Successful removed completed projects.”,workers\_tot,status\_list,completed\_projects,every\_project,project\_names)
* EXCEPT Exception as e
* RETURN (False,e,workers\_tot,status\_list,completed\_projects,every\_project,project\_names)

1. define function create\_project(status\_list,index,code\_of\_project,clients\_name,start\_date,expected\_end\_date,number\_of\_workers,project\_status,workers\_tot,every\_project,stats\_list,possible\_stats)

* TRY
* SET status\_list[index] TO status\_list[index] + 1
* SET project\_date TO [code\_of\_project,clients\_name,start\_date,expected\_end\_date,number\_of\_workers,project\_status,index]
* IF project\_status EQUALS “ongoing” and number\_of\_workers > workers\_tot
* RETURN (False,”There is not enough workers”, workers\_tot)
* IF project\_status EQUALS “ongoing”
* SET workers\_tot TO workers\_tot – number\_of\_workers
* APPEND code\_of\_project TO project\_names
* APPEND project\_data to all\_projects
* RETURN (True, “Successfully created a new project”, workers\_tot)
* EXCEPT Exception as e
* RETURN (False, e, workers\_tot)

1. define function update\_project\_details(status\_list,index,previous\_index,code\_of\_project,clients\_name,start\_date,expected\_end\_date,number\_of\_workers,project\_status,current\_workers,workers\_tot,previous\_project\_status)

* TRY
* IF project\_status EQUALS “ongoing”
* IF number\_of\_workers > workers\_tot + (current\_workers if previous\_project\_stautus EQUALS “ongoing” else 0)
* RETURN (False, “Workers chosen are too much”, workers\_tot)
* IF project\_status EQUALS “ongoing”
* SET workers\_tot TO workers\_tot – number\_of\_workers
* IF previous\_project\_status EQUALS “ongoing”
* SET workers\_tot TO workers\_tot + current\_workers
* SET status\_list[index] TO status\_list[index] + 1
* SET status\_list[previous\_index] TO status\_list[previous\_index] – 1
* SET project\_data TO [code\_of\_project,clients\_name,start\_date,expected\_end\_date,number\_of\_workers,project\_status,index]
* SET index TO project\_names.index(code\_of\_project)
* SET all\_projects[index] TO project\_data
* RETURN (True, “Project details updated successfully”, workers\_tot)
* EXCEPT Exception as e:
* RETURN (False,e,workers\_tot)

1. define function date\_vertification(msg)

* SET date TO INPUT(msg)
* SET splitted\_date TO date.split(date[2] if len(date) > 3 else “ “)
* IF len(splitted\_date) != 3
* OUTPUT “Enter a valid format of the date…!”
* SET month TO splitted\_date[0]
* SET date TO splitted\_date[1]
* IF month > 12
* OUTPUT “Enter a valid month! “
* RETURN date\_vertification(msg)
* IF date > 31
* OUTPUT “Enter a valid date !”
* RETURN date\_vertification(msg)
* RETURN date

1. define function project\_status\_vertification(msg,update\_status)

* SET project\_state TO INPUT(msg).replace(“ “,””).lower()
* IF project\_state NOT IN possible\_inputs:
* OUTPUT “The entered project status is incorrect”
* IF update\_status IS True
* SET statistics\_list[possible\_inputs.index(project\_state)] TO statistics\_list[possible\_inputs.index(project\_state)] + 1
* RETURN (project\_state,statistics\_list,possible\_inputs.index(project\_state))

1. define function project\_code\_vertification(msg,project\_codes)

* SET project\_code TO INPUT(msg)
* IF project\_code IN project\_codes
* OUTPUT “Project code already exists”
* RETURN project\_code\_vertification(msg,project\_codes)
* RETURN project\_code

1. define function check\_if\_int(msg)

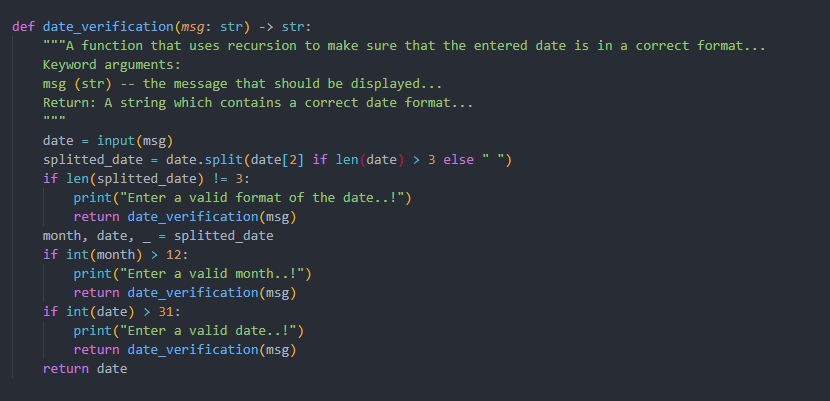
* TRY
* RETURN int(INPUT(msg)
* EXCEPT
* OUTPUT “The msg entered was not an integer”
* RETURN check\_if\_int(msg)

1. while execute

* SET choice TO menu(redirect\_choice,redirect\_to)
* SET redirect\_choice TO False
* SET redirect\_to TO None
* IF choice EQUALS “1”
* OUTPUT company\_name + “Add a new project”
* SET code\_of\_project TO project\_code\_vertification(“Project Code : “, project\_names)
* IF code\_of\_project EQUALS “0”
* CONTINUE
* SET clients\_name TO INPUT(“Clients Name : “)
* SET start\_date TO date\_vertification(“Start Date (MM/DD/YYYY) : ”)
* SET expected\_end\_date TO date\_vertification(“Expected end date (MM/DD/YYYY)”)
* SET number\_of\_workers TO check\_if\_int(“Numbers of Workers : “)
* SET project\_status, status\_list, index TO project\_status\_vertification()
* SET save to INPUT(“Do you want to save the project (Yes/No)”)
* IF save.upper() EQUALS “YES”
* SET execution\_status, response\_msg, workers TO create\_project(status\_list,index,code\_of\_project,clients\_name,start\_date,expected\_end\_date,number\_of\_workers,project\_status,workers,all\_projects,statistics\_list,possible-Inputs)
* OUTPUT response\_msg + execution\_status
* ELSE:
* OUTPUT “The project was \*not\* saved !”
* ELSEIF choice EQUALS “2”
* OUTPUT company\_name + “Remove completed project”
* SET code\_of\_project TO INPUT(“Project Code : “)
* IF code\_of\_project NOT IN project\_names
* OUTPUT “The project does not exist”
* CONTINUE
* SET save TO INPUT(“Do you want to save the project (Yes/No) ? “)
* IF save.upper() EQUALS “YES”
* SET Execution\_status,response\_msg,workers,status\_list,completed\_projects,every\_project, project\_names TO remove\_completed\_projects(code\_of\_project,all\_projects,workers,statistics\_list,completed\_projects,possible\_inputs)
* OUTPUT response\_msg + execution\_status
* ELSE
* OUTPUT “The project was not removed”
* ELSEIF choice EQUALS “3”
* OUTPUT company\_name + “Add new workers”
* SET new\_no\_of\_workers TO check\_if\_int(“Number Workers to Add : “)
* IF new\_no\_of\_workers < 0
* OUTPUT “Workers must be more than 0”
* IF save.upper() EQUALS “YES”
* SET workers TO workers + new\_no\_of\_workers
* ELSE
* OUTPUT “Workers were not added”
* ELSEIF choice EQUALS “4”
* OUTPUT company\_name + “Update Project Details”
* SET code\_of\_project TO INPUT(“Project Code : “)
* IF code\_of\_project NOT IN project\_names
* OUTPUT “There isn’t a project with the mentioned project code…!”
* IF code\_of\_project.replace(“ “,””) EQUALS “0”
* CONTINUE
* SET clients\_name TO INPUT(“Clients Name : “)
* SET start\_date TO date\_vertification(“Start Date (MM/DD/YYYY) : “)
* SET expected\_end\_date TO date\_vertification(“Excepected End Date (MM/DD/YYYY) : “)
* SET number\_of\_workers TO check\_if\_int(“Numbers of Workers : “)
* SET project\_status,status\_list,index TO project\_status\_vertification()
* SET save TO INPUT(“Do you want to update the project details (Yes/No)”)
* IF save.upper() EQUALS “YES”
* SET current\_workers,previous\_project\_status,previous\_index TO all\_projects[project\_names.index(code\_of\_project)][4:]
* SET execution\_status,response\_msg,workers TO update\_project\_details(status\_list,index,previous\_index,code\_of\_project,client\_name,start\_date,expected\_end\_date,number\_of\_workers,project\_status,current\_workers,workers,previous\_project\_status)
* OUTPUT response\_msg + execution\_status
* ELSE
* OUTPUT “The project was not updated”
* ELSEIF choice EQUALS “5”
* OUTPUT company\_name + “Project Statistics”
* FOR idx, item IN enumerate(possible\_inputs)
* OUTPUT “Number of “ + item + “projects : “ + statistics\_list[idx]
* OUTPUT “Number of Available Workers: “ + workers
* SET add\_project TO INPUT(“Do you want to add the project”)
* IF add\_project.upper() EQUALS “YES”
* SET redirect\_choice, redirect\_to TO True, “1”
* ELSEIF choice EQUALS “6”
* SET execute TO False
* ELSE
* OUTPUT “Please enter a valid choice!”

## Verification Functions

### date\_vertification()



Figure

`date\_vertification()` is a function that uses recursion to make sure the entered date is in the correct format. It has 1 argument which is `msg` which is the message that is displayed to the user. A string containing the correct date format is returned. The function works by first asking the user for a date, and then the data is split by the second character (3’rd letter), for example, “12**/**21/2008” The second element which is “/” will be used to split and it is taken in consideration that the string may be smaller than 3 letters so if it is then an empty string will be used. Then the length of the spliced list is checked and if it is not 3 then the `date\_vertification()` function calls itself (recursion). Then after that, the month and date are extracted from the list. Finally, the month and date are checked if they are higher than 12 and higher than 31 respectively, and if all the arguments are passed then the date is returned.

project\_status\_vertification()

A screenshot of a computer program

Description automatically generated

Figure

The `project\_status\_vertification()` function checks whether the project status that is entered is allowed and if not it uses recursion to make sure that the user enters the allowed status. It has 2 arguments which is `msg` which is the message that should be displayed to the user and `update\_status` which is a Boolean argument that if True the statistics\_list is updated with the status that was entered. First, the user has displayed a message which they need to respond to then the “ “ (blank spaces) are replaced with “and the entire message is lowered, then it is checked if the project\_state entered is not in the possible\_inputs list, if it is not then the function is calling itself (recursion) and if the project\_state is in the possible\_input the `statistics\_list` is updated and then the following is returned : (project\_state, statistics\_list, possible\_inputs.index(project\_state)) => (The project state, the statistics list that is used for the choice `5`, the index of the project state in the list possible\_inputs), an example would be: [“ongoing”,[2,1,5],1]

### project\_code\_vertification()

A screen shot of a computer code

Description automatically generated

Figure

`project\_code\_vertification()` is a function which uses recursion to make sure the project\_code entered does not exist already. The parameters are `msg` which is the message that should be displayed to the user and `project\_codes` which is the list of project\_codes where the function checks if the entered project code exists or not, and finally if the project code does not exist it is returned.

### check\_if\_int()

A screenshot of a computer program

Description automatically generated

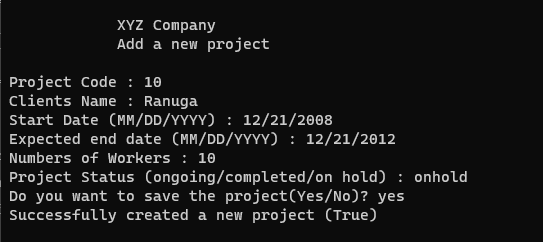
Figure

`check\_if\_int()` function uses recursion with having 1 parameter `msg` which is what is displayed to the user then the message is displayed and the function tries to return the message by trying to convert the inputted data into an integer and if an error is caused then a message saying “The msg entered was not an integer” is displayed and then the function calls itself (recursion).

## Main Functions

## Explanation

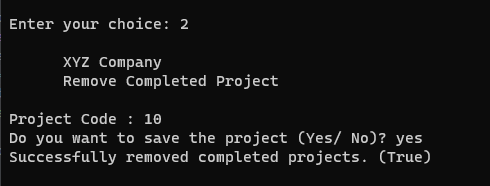
### Add a new project to existing projects.



Figure

When the user enters the choice `1` the program asks the user for the Project Code which is directed through a function `project\_code\_vertification()` which makes sure that the project code is unique, for the start date and expected end date, function `date\_vertification()` is used which makes sure the date entered is in the correct format of MM/DD/YYYY. When asking the user for the project status it is passed through a function `project\_status\_vertification()` which checks whether the status entered was either “ongoing”, “completed” or “on hold”.

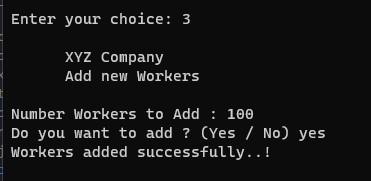
### Removing a Completed project from existing projects



Figure

When the user enters the choice `2` the program asks for the project code which needs to be removed, and it makes sure that the project code exists in the list `project\_names`.

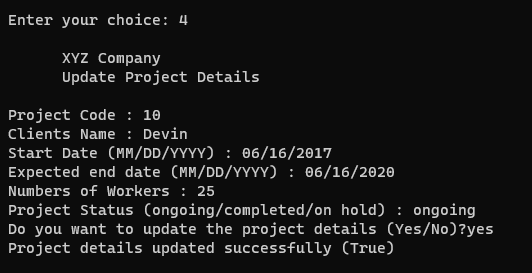
### Adding new workers to the available worker's group



Figure

When the user chooses the choice `3` the user is asked for the number of workers that they want to add and the function `check\_if\_it()` is used to make sure the user enters an integer value.

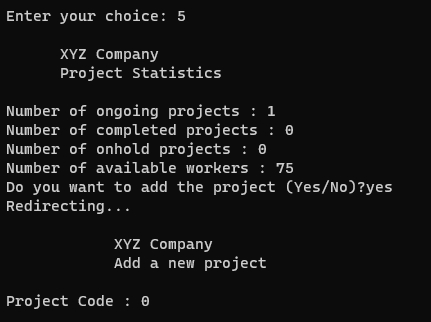
### Updating Details to an ongoing project



Figure

When the user chooses the choice `4` the user is asked for the project code and other details such as Clients Name, Start Date, Expected end Date, Number of workers, and Project Status which uses functions such as `date\_vertification()`, `check\_if\_int()` and `project\_status\_vertification()`, and this updates the data in the main list `all\_projects`

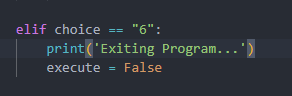
### Project Statistics



Figure

When the user chooses the choice `5` it displays the project statistics such as ongoing, completed, hold projects and the available workers. It also allows the user to be redirected to the 1st choice.

### Exiting the Program





When the user chooses the choice `6` then the program exits by changing the variable `execute` which is the condition that is given to the while loop in turn looping until `execute` is False.

# Assumptions

The listed below are assumptions that were made about the solution, are as listed below.

1. Unique Project Code

It was assumed that the project code must be unique throughout all other projects with either ongoing or on-hold status.

1. When removing a project, the current date should be stored.

It was assumed that the date on which the user removes a project needs to be stored as stated in the “DOC 333 CW Specification”, due to that reason the package “datetime” was imported to get the date on which the project is removed.

1. Only workers are assigned to ongoing projects.

It was assumed that only ongoing projects need to be assigned workers, the other statuses do not get assigned workers when they are created, but if they are updated to ongoing then workers are assigned to those projects.

1. The date format was assumed to be (MM/DD/YYYY)

It was assumed that the Date format that should be used is MM/DD/YYYY

1. There were 0 workers initially.

It is assumed that there are no workers initially when the program starts so it is required for the user to add workers before adding an `ongoing` project.

# Python Code

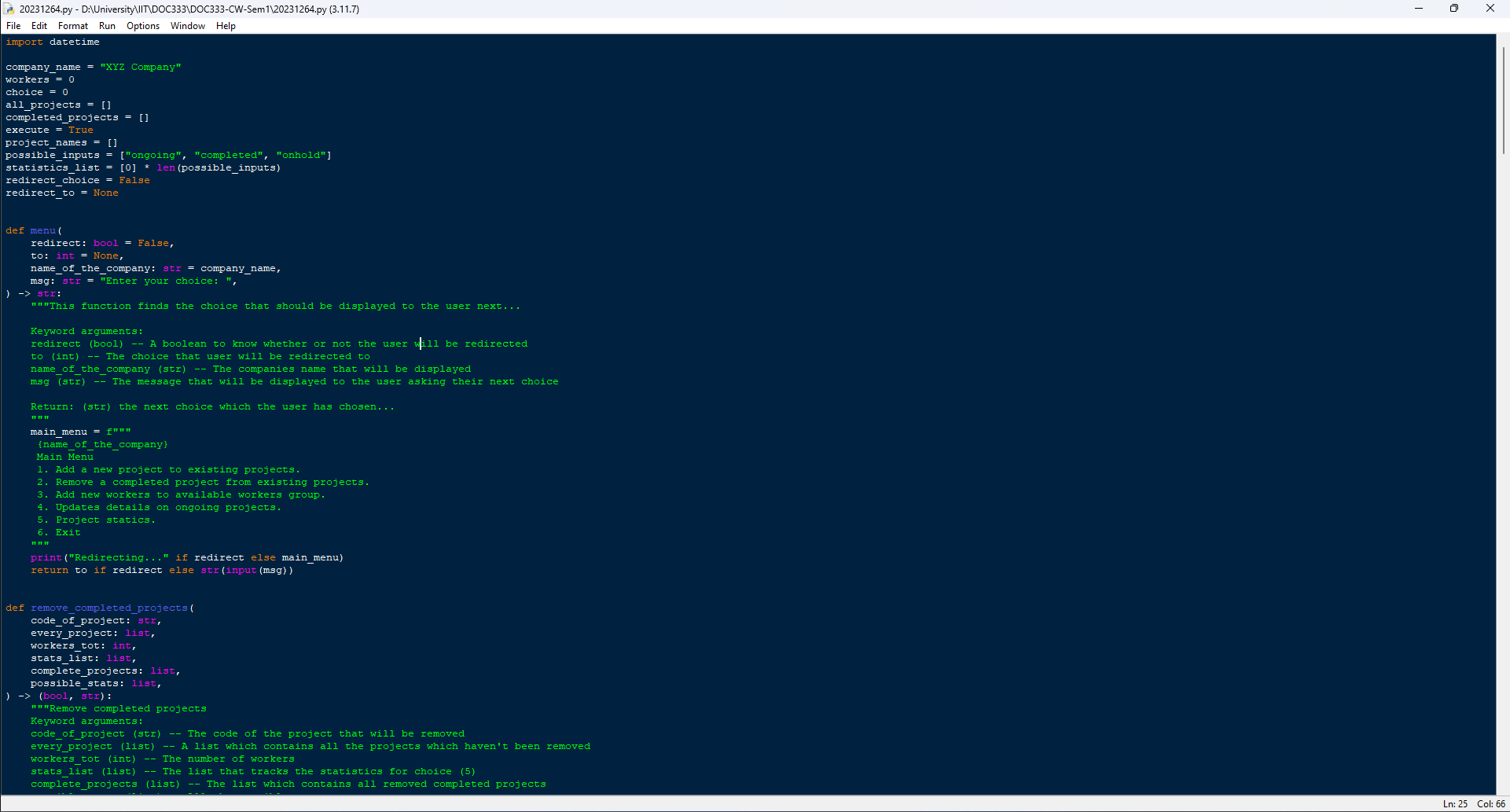


Figure 11

A screenshot of a computer

Description automatically generated

Figure 12

A screenshot of a computer

Description automatically generated

Figure 13

A screenshot of a computer

Description automatically generated

Figure 14

A screenshot of a computer program

Description automatically generated

Figure 15A screenshot of a computer

Description automatically generated

Figure 16A screenshot of a computer

Description automatically generated

Figure 17A screenshot of a computer

Description automatically generated

Figure 18

# Test Cases

## Add a New Project

### Test 1

The following test cases are for the 1st choice (“Add a new project to existing projects.”)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field | Input Entered | Expected Outcome | Actual Outcome | Results |
| Project Code | “0” | Exit and Go to the Main Menu | Exit and Go to the Main Menu | Pass |

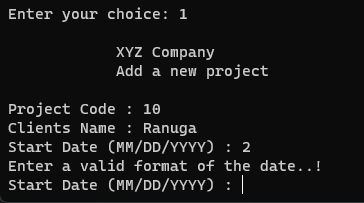


Figure

### Test 2

The following test cases are for the 1st choice (“Add a new project to existing projects.”)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field | Input Entered | Expected Outcome | Actual Outcome | Results |
| Project Code | “10” | Continue | Continue | Pass |
| Clients Name | “Ranuga” | Continue | Continue | Pass |
| Start Date | “2” | Error Raised, asking the user to enter the date again… | Error Raised, asking the user to enter the date again… | Pass |

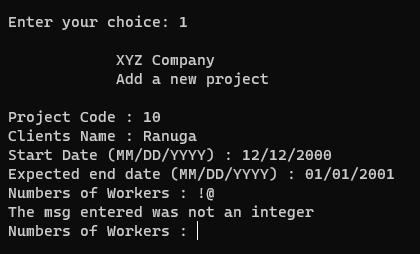


Figure

### Test 3

The following test cases are for the 1st choice (“Add a new project to existing projects.”)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field | Input Entered | Expected Outcome | Actual Outcome | Results |
| Project Code | “10” | Continue | Continue | Pass |
| Clients Name | “Ranuga” | Continue | Continue | Pass |
| Start Date | “12/12/2000” | Continue | Continue | Pass |
| Expected End Date | “01/01/2001” | Continue | Continue | Pass |
| Number of Workers | “!@” | “The msg entered was not an integer” | “The msg entered was not an integer” | Pass |



Figure

### Test 4

The following test cases are for the 1st choice (“Add a new project to existing projects.”)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field | Input Entered | Expected Outcome | Actual Outcome | Results |
| Project Code | “10” | Continue | Continue | Pass |
| Clients Name | “Ranuga” | Continue | Continue | Pass |
| Start Date | “12/12/2000” | Continue | Continue | Pass |
| Expected End Date | “01/01/2001” | Continue | Continue | Pass |
| Number of Workers | 10 | Continue | Continue | Pass |
| Project Status | “ongone” | “The entered project status is incorrect...” | “The entered project status is incorrect...” | Pass |

A computer screen with white text

Description automatically generated

Figure

### Test 5

The following test cases are for the 1st choice (“Add a new project to existing projects.”)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field | Input Entered | Expected Outcome | Actual Outcome | Results |
| Project Code | “10” | Continue | Continue | Pass |
| Clients Name | “Ranuga” | Continue | Continue | Pass |
| Start Date | “12/12/2000” | Continue | Continue | Pass |
| Expected End Date | “01/01/2001” | Continue | Continue | Pass |
| Number of Workers | 10 | Continue | Continue | Pass |
| Project Status | “on hold” | Continue | Continue | Pass |
| Save | “No” | “The project was \*not\* saved ..!” | “The project was \*not\* saved ..!” | Pass |

A screenshot of a computer program

Description automatically generated

Figure

### Test 6

The following test cases are for the 1st choice (“Add a new project to existing projects.”)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field | Input Entered | Expected Outcome | Actual Outcome | Results |
| Project Code | “10” | Continue | Continue | Pass |
| Clients Name | “Ranuga” | Continue | Continue | Pass |
| Start Date | “12/12/2000” | Continue | Continue | Pass |
| Expected End Date | “01/01/2001” | Continue | Continue | Pass |
| Number of Workers | 10 | Continue | Continue | Pass |
| Project Status | “on hold” | Continue | Continue | Pass |
| Save | “Yes” | “Successfully created a new project (True)” | “Successfully created a new project (True)” | Pass |

A computer screen with white text

Description automatically generated

Figure

## Remove a Project

### Assuming the project below has been added

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field | Input Entered | Expected Outcome | Actual Outcome | Results |
| Project Code | “10” | Continue | Continue | Pass |
| Clients Name | “Ranuga” | Continue | Continue | Pass |
| Start Date | “12/12/2000” | Continue | Continue | Pass |
| Expected End Date | “01/01/2001” | Continue | Continue | Pass |
| Number of Workers | 10 | Continue | Continue | Pass |
| Project Status | “on hold” | Continue | Continue | Pass |
| Save | “Yes” | “Successfully created a new project (True)” | “Successfully created a new project (True)” | Pass |

### Test 1

The following test cases are for the 2nd choice (“Remove Completed Project”)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field | Input Entered | Expected Outcome | Actual Outcome | Results |
| Project Code | “90” | “The project does not exist..!” | “The project does not exist..!” | Pass |

A screen shot of a computer

Description automatically generated

Figure

### Test 2

The following test cases are for the 2nd choice (“Remove Completed Project”)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field | Input Entered | Expected Outcome | Actual Outcome | Results |
| Project Code | “10” | Continue | Continue | Pass |
| Save | “No” | “The project was not removed…!” | “The v was not removed…!” | Pass |

A screen shot of a computer program

Description automatically generated

Figure

### Test 3

The following test cases are for the 2nd choice (“Remove Completed Project”)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field | Input Entered | Expected Outcome | Actual Outcome | Results |
| Project Code | “10” | Continue | Continue | Pass |
| Save | “Yes” | “Successfully removed completed projects.” | “Successfully removed completed projects.” | Pass |

A screen shot of a black screen

Description automatically generated

Figure

## Add new Workers

### Test 1

The following test cases are for the 3rd choice (“Add new Workers”)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field | Input Entered | Expected Outcome | Actual Outcome | Results |
| Number of Workers | -10 | “Workers must be more than 0..!” | “Workers must be more than 0..!” | Pass |

A black screen with white text

Description automatically generated

Figure

### Test 2

The following test cases are for the 3rd choice (“Add new Workers”)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field | Input Entered | Expected Outcome | Actual Outcome | Results |
| Number of Workers | “test” | “The msg entered was not an integer” | “The msg entered was not an integer” | Pass |

A screenshot of a computer screen

Description automatically generated

Figure

### Test 3

The following test cases are for the 3rd choice (“Add new Workers”)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field | Input Entered | Expected Outcome | Actual Outcome | Results |
| Number of Workers | 10 | Continue | Continue | Pass |
| Save | No | ” Workers were not added...!” | ” Workers were not added...!” | Pass |

A screenshot of a computer screen

Description automatically generated

Figure

### Test 4

The following test cases are for the 3rd choice (“Add new Workers”)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field | Input Entered | Expected Outcome | Actual Outcome | Results |
| Number of Workers | 10 | Continue | Continue | Pass |
| Save | Yes | “Workers added successfully...!” | “Workers added successfully...!” | Pass |

A screenshot of a computer screen

Description automatically generated

Figure

## Update Project Details

### Assuming the project below has been added

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field | Input Entered | Expected Outcome | Actual Outcome | Results |
| Project Code | “10” | Continue | Continue | Pass |
| Clients Name | “Ranuga” | Continue | Continue | Pass |
| Start Date | “12/12/2000” | Continue | Continue | Pass |
| Expected End Date | “01/01/2001” | Continue | Continue | Pass |
| Number of Workers | 10 | Continue | Continue | Pass |
| Project Status | “on hold” | Continue | Continue | Pass |
| Save | “Yes” | “Successfully created a new project (True)” | “Successfully created a new project (True)” | Pass |

### Test 1

The following test cases are for the 4th choice (“Update Project Details”)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field | Input Entered | Expected Outcome | Actual Outcome | Results |
| Project Code | 0 | Exit and Go to the Main Menu | Exit and Go to the Main Menu | Pass |

A black screen with white text

Description automatically generated

Figure

### Test 2

The following test cases are for the 4th choice (“Update Project Details”)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field | Input Entered | Expected Outcome | Actual Outcome | Results |
| Project Code | “10” | Continue | Continue | Pass |
| Clients Name | “Devin” | Continue | Continue | Pass |
| Start Date | “0000” | “Enter a valid format of the date…!” | “Enter a valid format of the date…!” | Pass |

A screen shot of a computer

Description automatically generated

Figure

### Test 3

The following test cases are for the 4th choice (“Update Project Details”)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field | Input Entered | Expected Outcome | Actual Outcome | Results |
| Project Code | “10” | Continue | Continue | Pass |
| Clients Name | “Devin” | Continue | Continue | Pass |
| Start Date | “01/01/2020” | Continue | Continue | Pass |
| Expected End Date | “007” | “Enter a valid format of the date…!” | “Enter a valid format of the date…!” | Pass |

A screen shot of a computer

Description automatically generated

Figure

### Test 4

The following test cases are for the 4th choice (“Update Project Details”)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field | Input Entered | Expected Outcome | Actual Outcome | Results |
| Project Code | “10” | Continue | Continue | Pass |
| Clients Name | “Devin” | Continue | Continue | Pass |
| Start Date | “01/01/2020” | Continue | Continue | Pass |
| Expected End Date | “12/12/2022” | Continue | Continue | Pass |
| Number of Workers | “test” | “The msg entered was not an integer” | “The msg entered was not an integer” | Pass |

A screen shot of a computer

Description automatically generated

Figure

### Test 5

The following test cases are for the 4th choice (“Update Project Details”)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field | Input Entered | Expected Outcome | Actual Outcome | Results |
| Project Code | “10” | Continue | Continue | Pass |
| Clients Name | “Devin” | Continue | Continue | Pass |
| Start Date | “01/01/2020” | Continue | Continue | Pass |
| Expected End Date | “12/12/2022” | Continue | Continue | Pass |
| Number of Workers | 25 | Continue | Continue | Pass |
| Project Status | “on done” | “The entered project status is incorrect…” | “The entered project status is incorrect…” | Pass |

A computer screen with white text

Description automatically generated

Figure

### Test 6

The following test cases are for the 4th choice (“Update Project Details”)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field | Input Entered | Expected Outcome | Actual Outcome | Results |
| Project Code | “10” | Continue | Continue | Pass |
| Clients Name | “Devin” | Continue | Continue | Pass |
| Start Date | “01/01/2020” | Continue | Continue | Pass |
| Expected End Date | “12/12/2022” | Continue | Continue | Pass |
| Number of Workers | 25 | Continue | Continue | Pass |
| Project Status | “on hold” | Continue | Continue | Pass |
| Save | “no” | “The project was not updated” | “The project was not updated” | Pass |

A computer screen with white text

Description automatically generated

Figure

### Test 7

The following test cases are for the 4th choice (“Update Project Details”)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field | Input Entered | Expected Outcome | Actual Outcome | Results |
| Project Code | “10” | Continue | Continue | Pass |
| Clients Name | “Devin” | Continue | Continue | Pass |
| Start Date | “01/01/2020” | Continue | Continue | Pass |
| Expected End Date | “12/12/2022” | Continue | Continue | Pass |
| Number of Workers | 25 | Continue | Continue | Pass |
| Project Status | “on hold” | Continue | Continue | Pass |
| Save | “yes” | “Project details updated successfully (True)” | “Project details updated successfully (True)” | Pass |

A screen shot of a computer

Description automatically generated

Figure

### Test 8

The following test cases are for the 4th choice (“Update Project Details”)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field | Input Entered | Expected Outcome | Actual Outcome | Results |
| Project Code | “58” | “There isn’t a project with the mentioned project code…!” | “There isn’t a project with the mentioned project code…!” | Pass |

A black screen with white text

Description automatically generated

Figure

## Project Statistics

### Assuming the project below has been added from Choice 3

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field | Input Entered | Expected Outcome | Actual Outcome | Results |
| Number of Workers | 100 | Continue | Continue | Pass |
| Save | “Yes” | “Workers added successfully” | “Workers added successfully” | Pass |

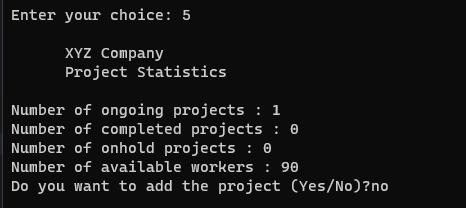
### Assuming the project below has been added from Choice 1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field | Input Entered | Expected Outcome | Actual Outcome | Results |
| Project Code | “10” | Continue | Continue | Pass |
| Clients Name | “Ranuga” | Continue | Continue | Pass |
| Start Date | “12/12/2000” | Continue | Continue | Pass |
| Expected End Date | “01/01/2001” | Continue | Continue | Pass |
| Number of Workers | 10 | Continue | Continue | Pass |
| Project Status | “on going” | Continue | Continue | Pass |
| Save | “Yes” | “Successfully created a new project (True)” | “Successfully created a new project (True)” | Pass |

### Test 1

The following test cases are for the 5th choice (“Project Statistics”)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field | Input Entered | Expected Outcome | Actual Outcome | Results |
| Add Project | “No” | Exit and Go to the Main Menu | Exit and Go to the Main Menu | Pass |



Figure

### Test 2

The following test cases are for the 5th choice (“Project Statistics”)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field | Input Entered | Expected Outcome | Actual Outcome | Results |
| Add Project | “Yes” | Redirect to Choice 1 | Redirect to Choice 1 | Pass |

A black screen with white text

Description automatically generated

Figure

## Exit

### Test 1

The following test cases are for the 6th choice (“Exit”)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field | Input Entered | Expected Outcome | Actual Outcome | Results |
| Choice | 6 | “Exiting Program…” | “Exiting Program…” | Pass |

A screen shot of a black screen

Description automatically generated

Figure