Survival Island (A Text Based Adventure Game) using Python

A Project Report for Industrial Training

Submitted by

Alapan Ghosh

Bishwajit Paul

Debarati Adhikari

Binu Show

Anirban Kumar

In the partial fulfillment for the award of the degree of

BCA

in

Stream

At

Ardent Computech Pvt. Ltd.

June-July 2018

CERTIFICATE FROM SUPERVISOR

This is to certify that ***Alapan Ghosh , Bishwajit Paul, Debarati Adhikari, Binu Shaw, Anirban Kumar*** s uccessfully completed the project titled " **Survival Island (A Text Based Adventure Game)using Python** " under my supervision during the period from June to July which is in partial fulfillment of requirements for the award of the Bachelors of Computer Applications and submitted to BCA Department of Techno India Hooghly.

S Sofikul Mullick
17/7/18

Signature of the Supervisor

Date:

Sofikul Mullick

Project Engineer

(Ardent Collaborations Pvt. Ltd.)

ACKNOWLEDGEMENT

The achievement that is associated with the successful completion of any task would be incomplete without mentioning the names of those people whose endless cooperation made it possible. Their constant guidance and encouragement made all our efforts successful.

We take this opportunity to express our deep gratitude towards our project mentor, **Sofikul Mullick** for giving such valuable suggestions, guidance and encouragement during the development of this project work.

Last but not the least we are grateful to all the faculty members of Ardent Computech Pvt. Ltd. for their support.

ABSTRACT

Text games are typically easy to write and require less processing power than games with graphics, and thus were more common from 1970 to 1990. However, terminal emulators are still in use today, and people continue playing **MUDs**(multi-user dungeon) and exploring interactive fiction. Many beginning programmers still create these types of games to familiarize themselves with a programming language,[1] and contests even now are held on who can finish programming a **roguelike** within a short time period, such as seven days.

While many of the earliest computer games relied on language parsing due to the command-line driven, teletype-terminal mainframe environments in which they were developed, the phrase "text-based" is taken to refer not to the user input (though generally keyboard-based) but rather to exclusive use of the fixed-width character display mode.

In this project, we tried to implement our knowledge in **Python** to build a fun text based adventure game

TABLE OF CONTENTS

CONTENTS **Page Number**

INTRODUCTION

Games are a popular medium of entertainment in the 21st century. In this game, we go back to era of classic text based RPGs, before visuals were a metric for judging games.

Survival Island is a text based adventure game where the player is stuck on a remote island and the objective is to survive until being rescued. The game tries to trick players into making decisions which seem logical but may not always work for their favour.

The game has multiple endings, though not varying greatly, rewards the player with quick success for taking risks. But not all risks are worth it.

We explore the different aspects of the project ahead.

Project Definition:

A text based game is basically a video game that uses text characters instead of bitmap or vector graphics. Text-based games were a popular form of interactive fiction in the 1980s.

Survival Island is a simple text based adventure game. The game is based on Python language .The game is about a person who is trapped in a island. In every stage users are given three opportunities to face the difficulties. Among the options user have to choose the right one for survive.

Project Objectives:

1.The main objective of this project is to make a fun game.

2.User have to think well during playing this game.

3.A text based game increase the patience of reading among the users.

4. Survival Island is actually based on practical logic.

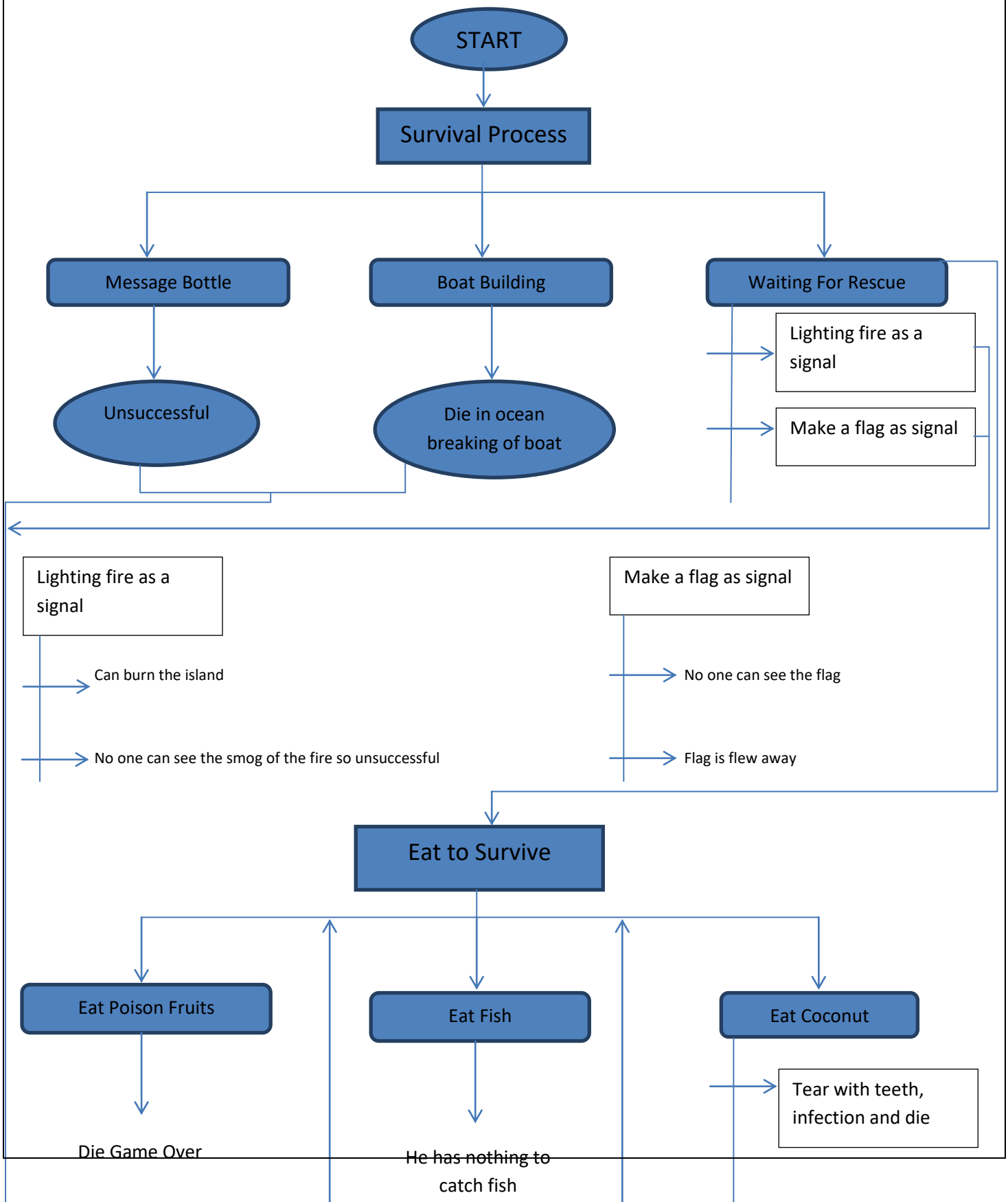
SYSTEM ANALYSIS

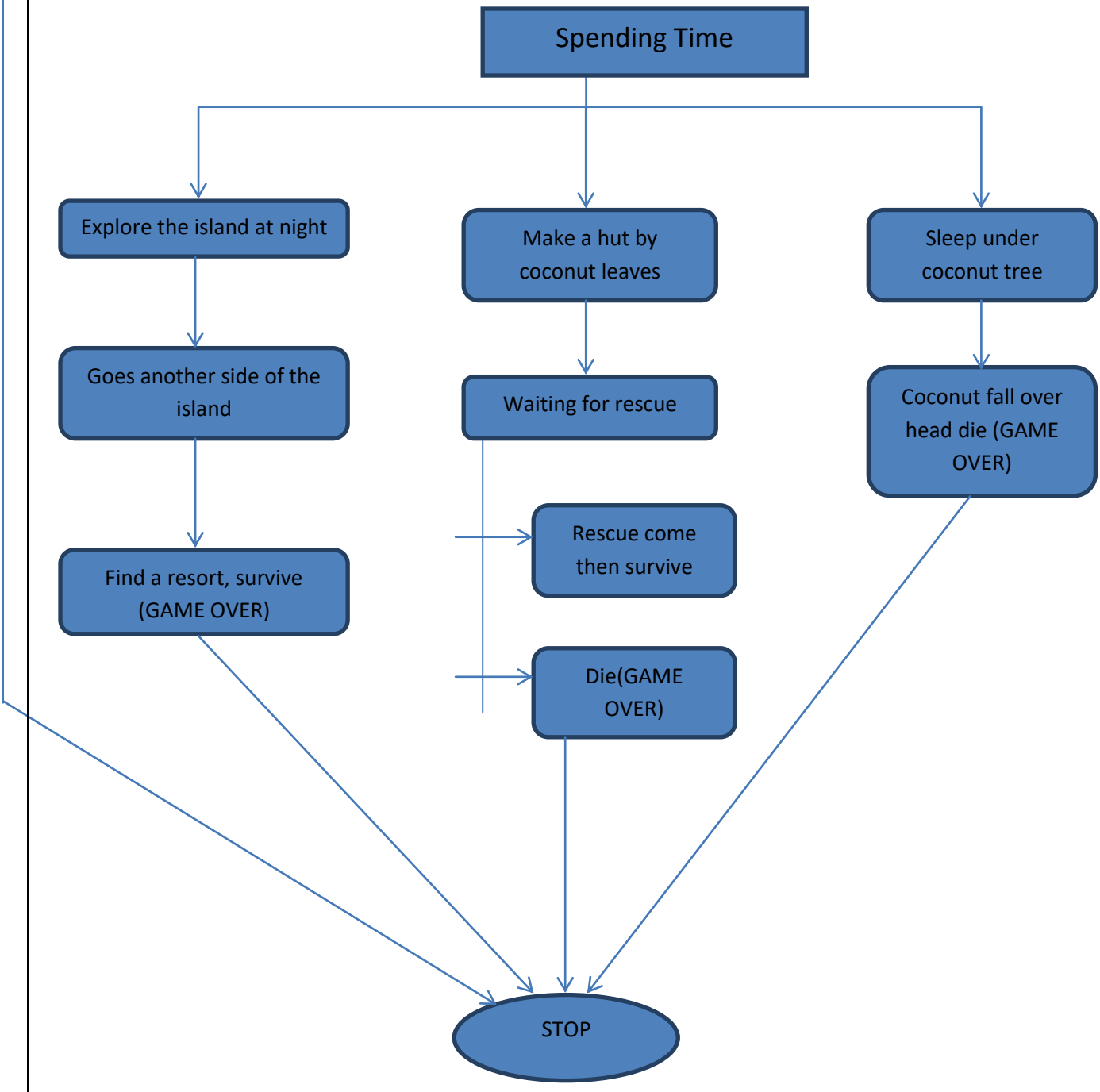
Feasibility Study:

This game is built on Python 3.6.5 which is free to download and use. The project was completed within a week and took less than 4 hrs to code.

The code uses functionalities from python's in built modules like **time**, **os** and **tkinter** which drastically decreased the development time and improved code portability.

Diagram:


START
Survival Process
Message Bottle
Boat Building
Waiting For Rescue
Unsuccessful
Die in ocean breaking of boat
Lighting fire as a signal
Make a flag as signal
Lighting fire as a signal
Can burn the island
No one can see the smog of the fire so unsuccessful
Make a flag as signal
No one can see the flag
Flag is flew away
Eat to Survive
Eat Poison Fruits
Eat Fish
Eat Coconut
Tear with teeth, infection and die
Die Game Over
He has nothing to catch fish


Spending Time
Explore the island at night
Make a hut by coconut leaves
Sleep under coconut tree
Goes another side of the island
Waiting for rescue
Coconut fall over head die (GAME OVER)
Rescue come then survive
Find a resort, survive (GAME OVER)
Die(GAME OVER)
STOP

HARDWARE AND SOFTWARE REQUIREMENTS

HARDWARE USED

1. Intel Core i5 7400(7th Gen, 3.5 GHz, 6M Cache)
2. 8 GB DDR4 DRAM
3. 1 TB HDD
4. NVIDIA GTX 1050ti GPU

SOFTWARE USED

1. Windows 10 (OS)
2. Python 3.6.5
3. Anaconda 3
4. Spyder

MINIMUM REQUIREMENTS FOR THIS PROJECT

1. Windows 7 or later (OS)
2. 1 GB RAM
3. 20 GB of total HDD space
4. Python 3.5 or later

TIME MODULE

We have used TIME in this project for giving timing ques to the program.

This module provides various time-related functions.

Although this module is always available, not all functions are available on all platforms. Most of the functions defined in this module call platform C library functions with the same name. It may sometimes be helpful to consult the platform documentation, because the semantics of these functions varies among platforms.

Functions:-

time.sleep(secs):

Suspend execution of the calling thread for the given number of seconds. The argument may be a floating point number to indicate a more precise sleep time. The actual suspension time may be less than that requested because any caught signal will terminate the sleep() following execution of that signal's catching routine. Also, the suspension time may be longer than requested by an arbitrary amount because of the scheduling of other activity in the system.

Changed in version 3.5: The function now sleeps at least secs even if the sleep is interrupted by a signal, except if the signal handler raises an exception.

OS MODULE

We have used OS in this project for giving system commands to shell for terminal operations.

This module provides a portable way of using operating system dependent functionality. If you just want to read or write a file see open(), if you want to manipulate paths, see the os.path module, and if you want to read all the lines in all the files on the command line see the fileinput module. For creating temporary files and directories see the tempfile module, and for high-level file and directory handling see the shutil module.

Functions:-

os.system():

Execute the command (a string) in a subshell. This is implemented by calling the Standard C function system(), and has the same limitations. Changes to sys.stdin, etc. are not reflected in the environment of the executed command. If *command* generates any output, it will be sent to the interpreter standard output stream.

On Unix, the return value is the exit status of the process encoded in the format specified for wait(). Note that POSIX does not specify the meaning of the return value of the C system() function, so the return value of the Python function is system-dependent.

On Windows, the return value is that returned by the system shell after running *command*. The shell is given by the Windows environment variable COMSPEC: it is usually **cmd.exe**, which returns the exit status of the command run; on systems using a non-native shell.

Tkinter MODULE

Tkinter is a Python binding to the Tk GUI toolkit. It is the standard Python interface to the Tk GUI toolkit, and is Python's de facto standard GUI. Tkinter is included with the standard Microsoft Windows and Mac OS X install of Python.

The name *Tkinter* comes from *Tk interface*. Tkinter was written by Fredrik Lundh.

As with most other modern Tk bindings, Tkinter is implemented as a Python wrapper around a complete Tcl interpreter embedded in the Python interpreter. Tkinter calls are translated into Tcl commands which are fed to this embedded interpreter, thus making it possible to mix Python and Tcl in a single application.

Python 2.7 and Python 3.1 incorporate the "themed Tk" ("ttk") functionality of Tk 8.5. This allows Tk widgets to be easily themed to look like the native desktop environment in which the application is running, thereby addressing a long-standing criticism of Tk (and hence of Tkinter).

SOURCE CODE

```
import time
import os
c1=0
def s1():
    os.system('cls')
    print('Your ship wrecked and you washed ashore on a remote island. You wake up and find yourself alone,with nothing.\n You have survive.You sure you up to this task?')
    print('You find a bottle and some paper and a pen along the shore, some logs and rope. What do you do?')
    print('\n1. Write a message in the bottle and throw it in the ocean')
    print('2. Try building a boat and escaping the island')
    print('3. Keep waiting for someone to come and rescue you\n\n')
    i=int(input('>>>'))
    if i==1:
        os.system('cls')
        print('Good job. Now wait another 100 years before someone finds the bottle and find you')
        time.sleep(3)
        s1()
    elif i==2:
        os.system('cls')
        print('You build a raft with the logs and set out into the ocean')
        time.sleep(2)
        print('Suddenly a storm comes.')
        time.sleep(2)
        print('The ocean current destroys your raft and you drown')
```

```
        time.sleep(2)
        go(1)
    elif i==3:
        s2()
    else:
        s1()
def s2():
    global c1
    os.system('cls')
    print('You have been waiting for hours now. Why not try some other things while you wait for being rescued?\n')
    print('You can light a fire with sticks and stones, use your shirt to make flag, or shout and scream like an idiot.\nWhat would you do?')
    print('\n1. Let\'s get it blazing')
    print('2. I can go shirtless')
    print('3. I am an idiot\n\n')
    i=int(input('>>>'))
    if i==1:
        c1=c1+1
        os.system('cls')
        print('A storm came and put out the fire. Maybe something else?')
        time.sleep(3)
        if c1 % 3==0:
            s3()
        else:
            s2()
    elif i==2:
        c1=c1+1
        os.system('cls')
```

```
        print('A storm came and blew away the flag. Congrats, now you are both flagless and shirtless')
        time.sleep(3)
        if c1 % 3==0:
            s3()
        else:
            s2()
    elif i==3:
        c1=c1+1
        os.system('cls')
        print('Well, nobody can see you. Why not do something useful?')
        time.sleep(3)
        if c1 % 3==0:
            s3()
        else:
            s2()
    else:
        s2()
def s3():
    os.system('cls')
    print('Well all your attempts certainly didnt work and now you are tired and hungry.\nYou see some berries in the bushes and a lot of coconut trees around. There are also fish in the sea')
    print('So what do you want for lunch?\n')
    print('1. Lets have some berries')
    print('2. I\'ll go fishing')
    print('3. Coconuts are the best\n\n')
    i=int(input('>>>'))
    if i==1:
        os.system('cls')
```

```
        print('The berries were poisonous. Didnt you watch Discovery Channel?')
        time.sleep(3)
        go(1)
    elif i==2:
        os.system('cls')
        print('You have no experience in catching fish. Try something else')
        time.sleep(3)
        s3()
    elif i==3:
        s4()
    else:
        s3()
def s4():
    os.system('cls')
    print('You see a lot of coconut trees around. You climb up and get a coconut. Now you have to break into it\nWhat would you do?\n')
    print('1. Try hitting the coconut on a rock and break it')
    print('2. Try to tear the coconut with your teeth')
    print('3. Get one more coconut from the tree\n\n')
    i=int(input('>>>'))
    if i==1:
        s5()
    elif i==2:
        os.system('cls')
        print('You tried to bite the coconut hard and a tooth fell off. The would gave you an infection and you died')
        time.sleep(3)
        go(1)
    elif i==3:
```

```
        os.system('cls')
        print('After getting the coconut as tried to get down from the tree you foot slipped and you fell')
        go(1)
    else:
        s4()
def s5():
    os.system('cls')
    print('The coconut was a good choice. It has enough water and nutrients.\nNow its nearing dusk. The sun is about to set. What do you want to do?\n')
    print('\n1. Go explore the island')
    print('2. Have some rest and build a shelter\n\n')
    i=int(input('>>>'))
    if i==1:
      s6()
    elif i==2:
        s7()
    else:
        s5()
def s6():
    os.system('cls')
    print('You are exploring the island and its now night. You are walking in one direction while suddenly you hear a faint sound from another direction.\nWhat do you do?')
    print('\n1. Investigate the source of the sound')
    print('2. Keep walking')
    print('3. Go back\n\n')
    i=int(input('>>>'))
    if i==1:
        os.system('cls')
```

```
        print('You keep going and see some lights...')
        time.sleep(2)
        print('As you move closer, you see a magnificent resort, a luxury beach and lots of people')
        time.sleep(2)
        print('You have finally reached some civilisation. Congrats, you have been rescued')
        time.sleep(3)
        go()
    elif i==2:
        os.system('cls')
        print('You keep walking in the dark, and suddenly fall off a cliff')
        time.sleep(3)
        go(1)
    elif i==3:
        s5()
    else:
        s6()
def s7():
    os.system('cls')
    print('You realised you need some sleep after such an adventurous day. What do you do?\n')
    print('1. Make a small shelter using the coconut leaves')
    print('2. Dont make a shelter and sleep under the open sky')
    print('3. Sleep among the trees for shelter\n\n')
    i=int(input('>>>'))
    if i==1:
        s8()
    elif i==2:
        s8()
```

```
    elif i==3:
        os.system('cls')
        print('A coconut fell while you slept and cracked your skull. Never sleep under coconut trees')
        time.sleep(3)
        go(1)
def s8():
    os.system('cls')
    print('You wake up next morning and see helicoptors flying towards the island')
    time.sleep(2)
    print('They have come to rescue you. Glad you survived this long')
    time.sleep(2)
    print('By the way, did you know that there was a resort on the other side of the island?')
    time.sleep(2)
    print('Perhaps if you explored the island a bit more, you would have been rescued faster')
    time.sleep(2)
    go()
def go(d=0):
    if d==0:
        os.system('cls')
        print('YOU HAVE SUCCESSFULLY SURVIVED THE ISLAND.\n CONGRATULATIONS...!!!')
        time.sleep(5)
    else:
        os.system('cls')
        print('SORRY...YOU DIED...!!!')
        time.sleep(5)
    print('\n\n\n\nWOULD YOU LIKE TO REPLAY?')
```

```
    r=input('>>>')
    if r=='Y' or r=='y':
        s1()
s1()
```

FUTURE SCOPE AND PLAN

In this project, we try to create a fun little adventure game reminiscent of the bygone era. The story has some version of a branching narrative, where the objective can be completed in multiple ways. In a future version, we hope to have a much deeper RPG-like approach in story-telling.

We also plan to add a scoring system for players to view and compare their scores.

All future updates of the game can be freely downloaded from:

https://github.com/AlapanGhosh/tbag

LIMITATIONS

This project relies heavily on the command line shell on Windows and would not execute perfectly otherwise. Currently, the project is only supported on Windows, although it is possible to port it to other platforms.

It also needs python preinstalled to run, which is a minor setback.

SUMMARY

Python as a development platform shows great potential for the future. It has been slowly taking over multiple domains of computer applications and entertainment is no exception.

Several games so far have been made with the help of Python, including some AAA titles made entirely of Python. In this project we try to sample the potential of the language.

Due to Python's flexibility, this project can be easily ported to other platforms like Linux and the Mac OS.

BIBLIOGRAPHY

- WWW.PYTHON.ORG
- WWW.W3SCHOOLS.COM/PYTHON
- WWW.TUTORIALSPOINT.COM/PYTHON
- https://docs.python.org/3/index.html