BSc (Hons) Artificial Intelligence and Data Science

Module: CM 1601
Individual Coursework
Module Leader : Sachinthani Perera

RGU Student ID : 2331416

IIT Student ID : 20230177

Student Name : M.A.Wasif Asi

Abstract

This Coursework is about creating a python program to Horse Race . In here we have to input maximum 20 Horses into 4 groups(Group A, Group B, Group C, and Group D). So, 5 horses Each group.

In here user can run the code by inputting at least 1 horse for each group. Else code will ask user to add horses for groups.

Contant Table

List of Figures

1.0 Console Menu

In This Python Course Work I used module Method for Easy and Error Handling. It has two Global variables. All the main codes are in a while Loop , So Everything will be repeated until use want to exit.

At first this menu will print what to input For what to do. Then Ask User's Choice and that choice will convert Upper case.

If user input **AHD** that will go to adding horses' details part. User have to add horse details one by one when the same will repeating . User can add maximum 20 horses' detail.

If user want to change the horse details, then user can input **UHD**.

If user want to delete the horse details, then user can input **DHD.**

By adding **VHD** user can view shorted horse details by horse ids That are in the List.

To save the horse details in a text file user have to input **SHD.**

After all of those users can start the game menu By inputting **START**.

Inside the game menu by adding **SDD** user can select one horse randomly from Each Group and that will added to selected Horse list.
By adding **WHD** user can random winning Time for Each selected Horses and find out which horse has short time and that will print out.
By adding **VWH** User can visualize the Race time by * mark .
User can go back to game menu by adding **STOP.**

By adding **ESC** user can Exit The whole game.

1.1 Python Code

```
#Importing Files
from AHD import add_horse
from UHD import update_horse
from DHD import delete_horse
from VHD import view_horses
from SHD import save_to_file
from SDD import select_horses_for_race
from WHD import Winning_Horse
from VWH import Visualize_Winning

# Global Variable Creation
total_horse_details=[]
selected_horses=[]

# Main console Menu
while True:
    print("\n===== Horse Race Event Menu =====")
    print("1. Type AHD for adding horse details.")
    print("2. Type UHD for updating horse details.")
    print("3. Type DHD for deleting horse details.")
    print("4. Type VHD for viewing the registered horses\' details table.")
    print("5. Type SHD for saving the horse details to the text file.")
    print("6. Type START for Start the Game \n")
    print("7. Type ESC to exit the program.")

    choice = input("\nEnter your choice: ").upper()

    if choice == "AHD":
        add_horse(total_horse_details)
    elif choice == "UHD":
        update_horse(total_horse_details)
    elif choice == "DHD":
        delete_horse(total_horse_details)
    elif choice == "VHD":
        view_horses(total_horse_details)
    elif choice == "SHD":
        save_to_file(total_horse_details)
    elif choice == "START":

        # Clearing The List
        selected_horses.clear()

        # Starting the Game Menu
        # When The Game Started  You Have to Stop the Game Menu To do Previous Choices
        while True:
            print(" \n\t---Game is Started ---")
            print("1. Type SDD for selecting four horses randomly for the major round.")
            print("2. Type WHD for displaying the Winning horses\' details.")
```

```
        print("3. Type VWH for Visualizing the time of the winning horses.")
        print("4. Type STOP  the Game And Entering the Menu . ")

        choice = input("\nEnter your choice: ").upper()

        if choice == "SDD":
            select_horses_for_race(total_horse_details,selected_horses)
        elif choice == "WHD":
            Winning_Horse(selected_horses)
        elif choice == "VWH":
            Visualize_Winning(selected_horses)
        elif choice == "STOP":                          # Getting Out Of the Game Menu
            print("\nGetting Out The Menu")
            break
        else:
            print("\nInvalid choice. Please enter a valid option.")

elif choice == "ESC":                                   # Getting Out Of the Program
    print("Exiting the program. Goodbye!")
    break
else:
    print("\nInvalid choice. Please enter a valid option.")
```

1.2 Flow Chart

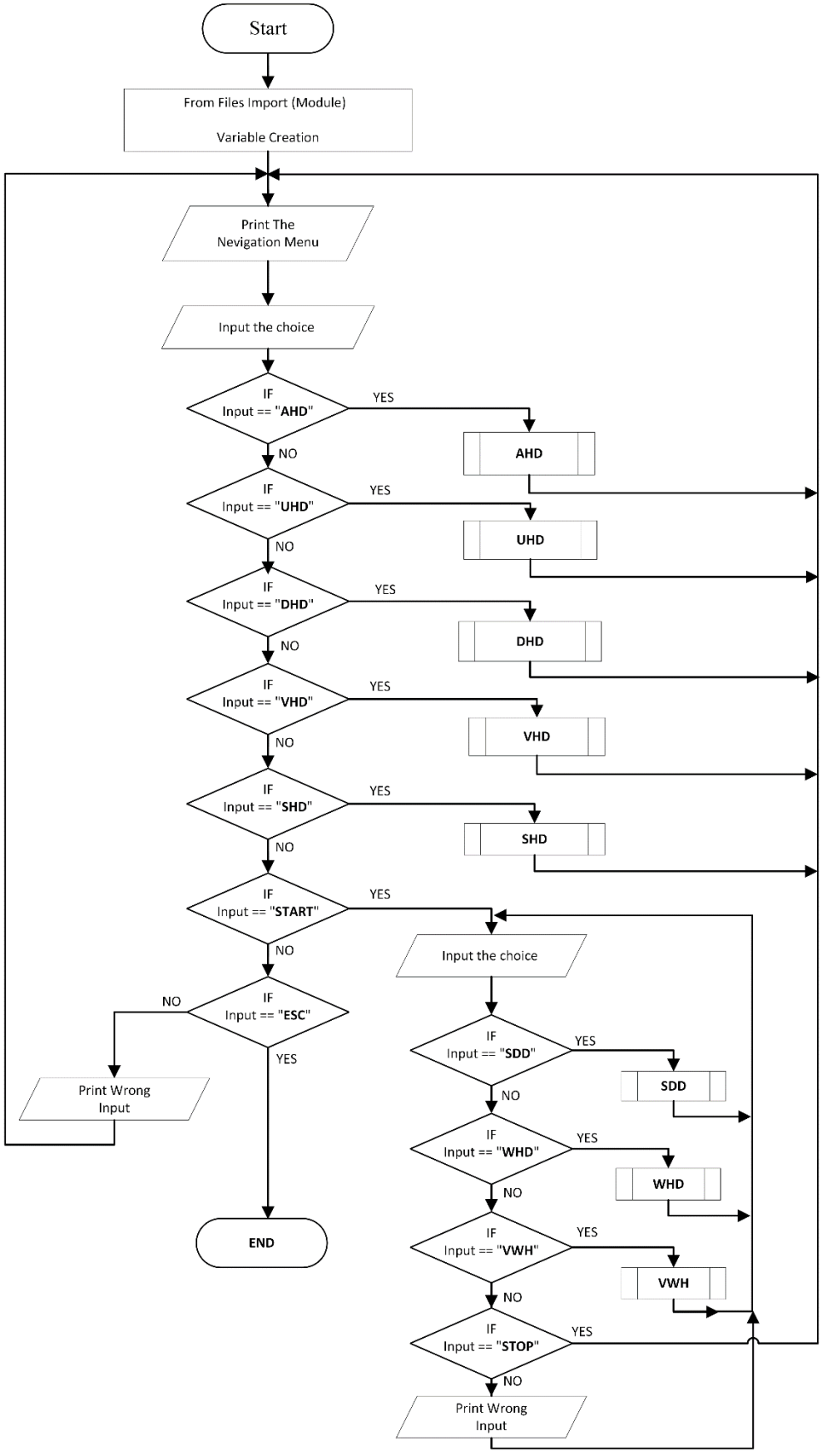

Figure 1 Flowchart For Console Menu

1.3 Screenshot Of The Output.

```
Enter your choice: start

        ---Game is Started ---
1. Type SDD for selecting four horses randomly for the major round.
2. Type WHD for displaying the Winning horses' details.
3. Type VWH for Visualizing the time of the winning horses.
4. Type STOP  the Game And Entering the Menu .

Enter your choice:
```

Figure 4 Menu program when running

```
Enter your choice: 12

Invalid choice. Please enter a valid option.

        ---Game is Started ---
1. Type SDD for selecting four horses randomly for the major round.
2. Type WHD for displaying the Winning horses' details.
3. Type VWH for Visualizing the time of the winning horses.
4. Type STOP  the Game And Entering the Menu .

Enter your choice:
```

Figure 3 If input is wrong

```
===== Horse Race Event Menu =====
1. Type AHD for adding horse details.
2. Type UHD for updating horse details.
3. Type DHD for deleting horse details.
4. Type VHD for viewing the registered horses' details table.
5. Type SHD for saving the horse details to the text file.
6. Type START for Start the Game

7. Type ESC to exit the program.

Enter your choice:
```

Figure 2 Starting menu For the game

2.0 Adding Horse Details (AHD)

By input AHD we can enter to AHD function. This Function has four Local variables to count how many horses in in each group. One group can have 5 horses maximum .

At first This Function automatically check how many horses there in each group before adding horses into the groups. After that it's check for total horses . If it's under 20 it will user to add horses ,else it will say 20 horses reached .

Then it will ask for an id to the horse . Id will be an integer. If user inputs Rather than an integer it will automatically say add an integer and ask user to input. Also, this Code check That this horse id is already there or not. If it is there It will ask for the same input again.

Then it will ask the name of the horse. Name can be only alphabetics. If the user inputs an integer the name input will be ask again.

Then it will ask the name of the Jockey. Also, this name can be only alphabetics. If the user inputs an integer the Jockey name input will be ask again.

Then it will ask the age of the horse .Age can be only integer. If the user inputs an alphabetics or characters, the age input will be asked again.

Then it will ask the Breed of the horse. Breed can be only alphabetics. If the user inputs an integer the breed input will be ask again.

Then it will ask for how many wins that horse has and in how many races . both are integers . If input is alphabetic Same question will repeat again. Then this function will merge both inputs into one string. Also wins can't be higher than races .if that happened same question will be repeated.

After all of those input code will ask for the group . Also, it will make a dictionary as horse with keys and values (inputs).

Then it will check for the availability of the groups . If the group that entered by user is available, then the horse details will append the total horse details . if not group in put will be ask again and again until user found an available group. If the entered Groups are not within A,B,C and D a message will pop up and say this input is wrong .

2.1 Python Code

```
def add_horse(total_horse_details):

    # Creating Local Variable
    Group_A_count = 0
    Group_B_count = 0
    Group_C_count = 0
    Group_D_count = 0

    # Counting how Many Horses in the Group
    for horse in (total_horse_details):

        if (horse['group']=="A"):
            Group_A_count+=1

        if (horse['group']=="B"):
            Group_B_count+=1

        if (horse['group']=="C"):
            Group_C_count+=1

        if (horse['group']=="D"):
                Group_D_count+=1


    #Seting a limit for totel horses
    if len(total_horse_details)<20:

        # Getting Horse id , validating and Duplication handling
        while True:
            horse_id = input("\nEnter Horse ID: ")
            if horse_id.isdigit():
                for horse in total_horse_details:
                    if horse['horse_id']== horse_id:
                        print("\nId is already there\n")
                        break
                else:
                    break
            else:
                print("\nInvalid input. Horse ID must be an integer.")


        # Validating Horse Name as Alphabetic
        while True:
            horse_name = input("Enter Horse Name: ")
            if  horse_name.isalpha():
                break
            else:
                print("\nInvalid input. Horse Name must contain only alphabetic characters.\n")
```

```
# Validating Jockey Name as Alphabetic
while True:
    jockey_name = input("Enter Jockey Name: ")
    if  jockey_name.isalpha():
        break
    else:
        print("\nInvalid input. Jockey Name must contain only alphabetic characters.\n")


# Validating Age as Integer
while True:
    age = input("Enter Age: ")
    if age.isdigit():
        break
    else:
        print("\nInvalid input. Age must be an integer.\n")


# Validating Breed Name as Alphabetic
while True:
    breed = input("Enter Breed: ")
    if  breed.isalpha():
        break
    else:
        print("\nInvalid input. Breed must contain only alphabetic characters.\n")


# Asking for race Record
while True:
    Wins = input("Enter How Many Wins: ")
    Total_Races = input("Enter How Many Matches: ")
    if  Wins.isdigit() and Total_Races.isdigit():
        if (int(Wins)<=int(Total_Races)):
            break
        else:
            print ("\nWins Can't be higher than Totel Races")
            continue
    else:
        print("\nInvalid input. Wins and Races must be an integer\n")

race_record =f"{Wins} Wins in {Total_Races} races"
```

```
    # Checking availabily for groups and adding horse for each group
    # If the Group is full You can Add to Other Groups
    while True:

        group = str(input("Enter Group (A, B, C, D): ")).upper()

        horse = {'horse_id': horse_id, 'horse_name': horse_name, 'jockey_name': jockey_name,
                'age': age, 'breed': breed, 'race_record': race_record, 'group': group}


        if (group == "A"):
            if Group_A_count < 5 :
                total_horse_details.append(horse)
                print("\nHorse details Added Successfully ")
                break
            else:
                print("\nA is full\n")

        elif (group == "B"):
            if Group_B_count < 5:
                total_horse_details.append(horse)
                print("\nHorse details Added Successfully ")
                break
            else:
                print("\nB is full\n")

        elif (group == "C"):
            if Group_C_count < 5:
                total_horse_details.append(horse)
                print("\nHorse details Added Successfully")
                break
            else:
                print("\nC is full\n")

        elif (group == "D"):
            if Group_D_count <  5:
                total_horse_details.append(horse)
                print("\nHorse details Added Successfully ")
                break
            else:
                print("\nD is full\n")

        else:
            print("\nWroung Group Input")

# If Limit is Reached You can add More Horses
else:
    print("\nAlready There were Totaly 20 Horses. You can't add more. ")
```

2.2 Flow Chart

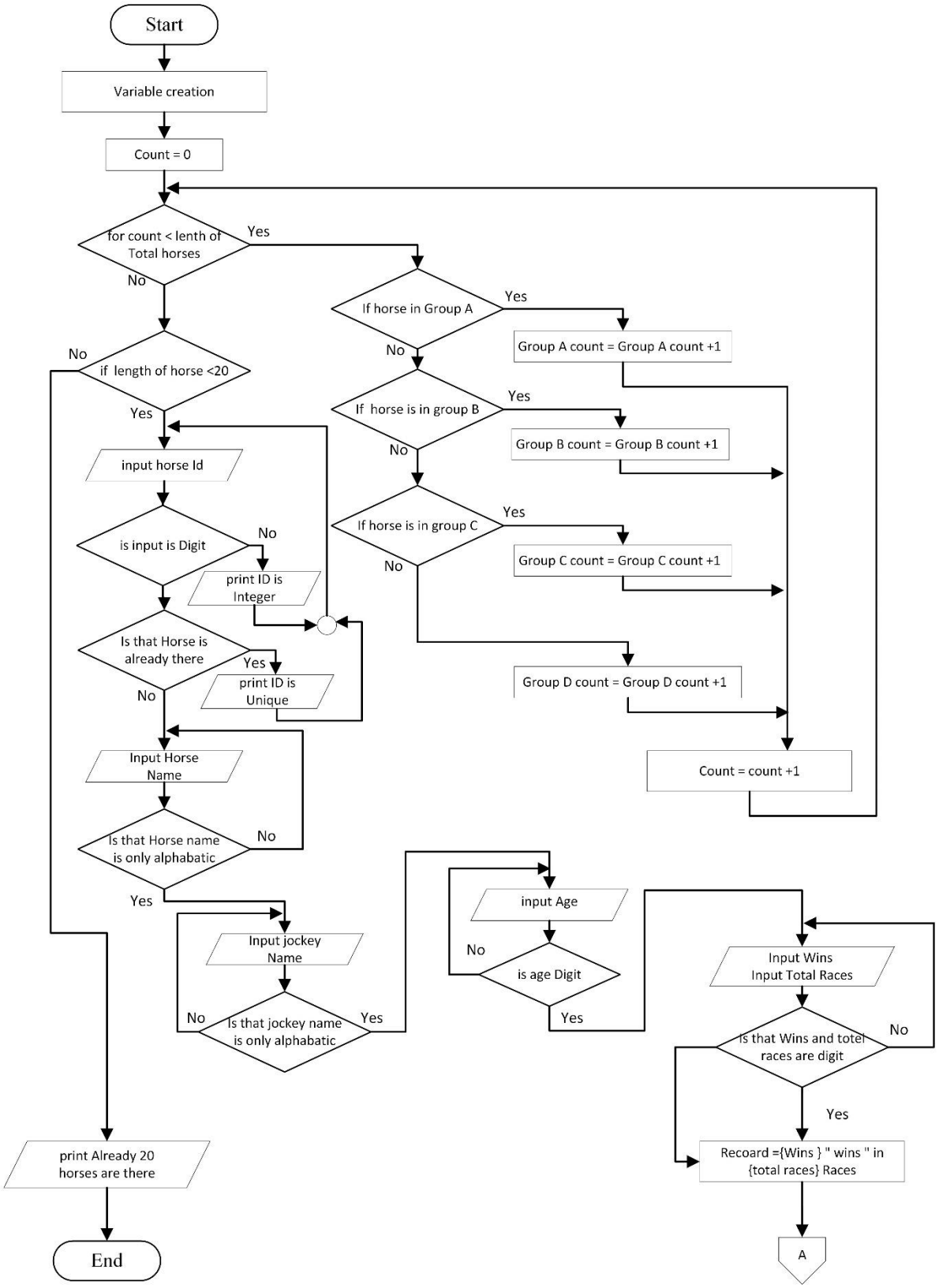

Figure 5 Flowchart 1 for AHD

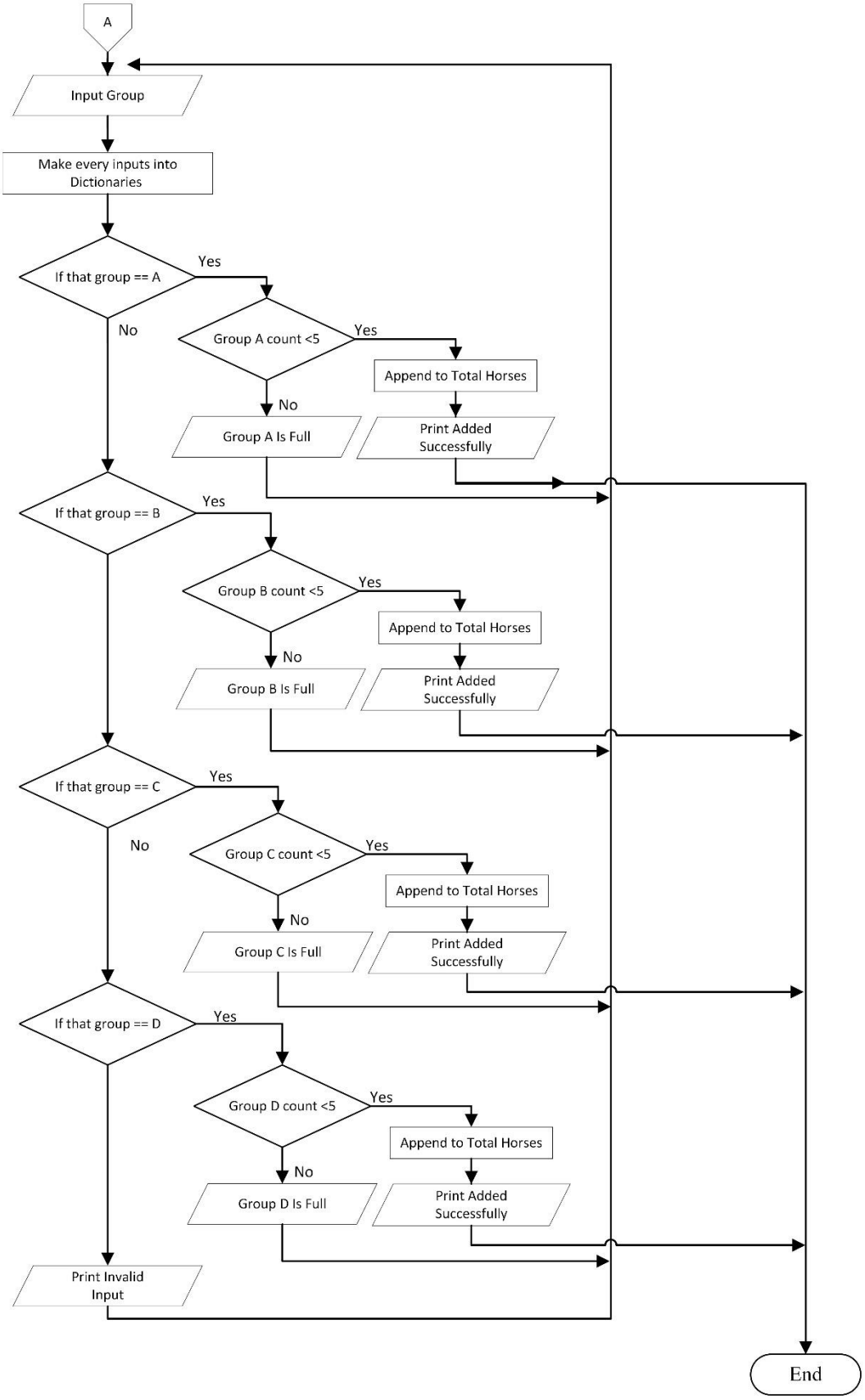

Figure 6 Flowchart 2 for AHD

2.3 Screenshot Of The Output.

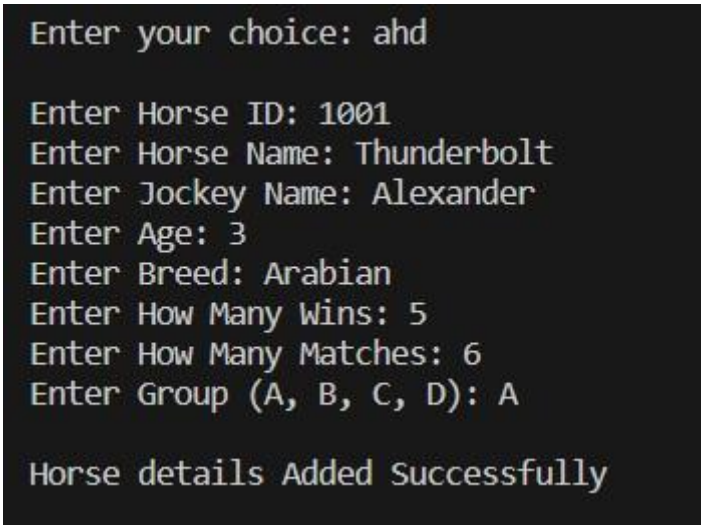

Figure 7 Adding horse Details.

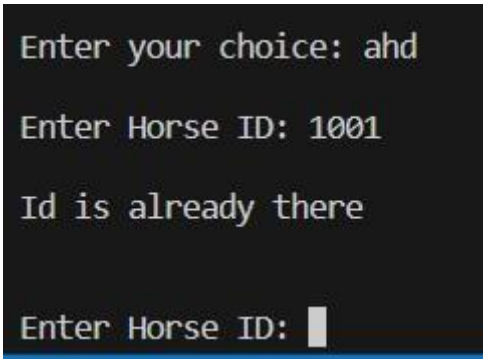

Figure 8 Duplication Checking

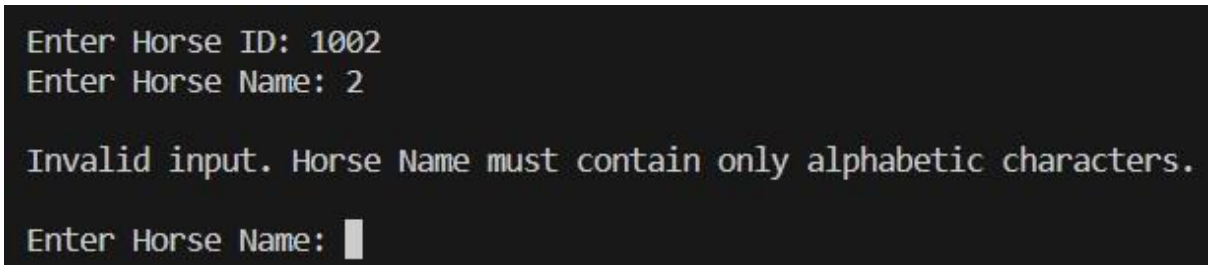

Figure 9 Alphabetic Or Integer checking

3.0 Updating Horse Details (UHD)

By entering to this function user can change details of the horse but Not the horse ID.

This function has four local variables . This UHD part is exact same as AHD part.

At first if will Count the count of each group. The ask for the Horse id . It will check that horse with that horse id is in the total horse details list or not . If it's not there a message will pop up as " Horse not found ! ". If that horse details are there then the group count of that horse will be reduced by one and then question will ask again.

Then ask input updated horse name. Name can be only alphabetics. If the user inputs an integer the name input will be ask again.

Then it will ask updated the name of the Jockey. Also, this name can be only alphabetics. If the user inputs an integer the Jockey name input will be ask again.

Then it will ask the updated age of the horse .Age can be only integer. If the user inputs an alphabetics or characters, the age input will be asked again.

Then it will ask the updated Breed of the horse. Breed can be only alphabetics. If the user inputs an integer the breed input will be ask again.

Then it will ask for updated how many wins that horse has and in how many races . both are integers . If input is alphabetic Same question will repeat again. Then this function will merge both inputs into one string. Also wins can't be higher than races .if that happened same question will be repeated.

After all of those input code will ask for the updated group. Also, it will make a dictionary as horse with keys and values (inputs).

Then it will check for the availability of the groups. If the group that entered by user is available, then the horse details will append the total horse details . if not group in put will be ask again and again until user found an available group. If the entered Groups are not within A,B,C and D a message will pop up and say this input is wrong .

3.1 Python Code

```
def update_horse(total_horse_details):

    # Creating Local Variable
    Group_A_count=0
    Group_B_count=0
    Group_C_count=0
    Group_D_count=0

    # Counting How many horses in each group
    for horse in (total_horse_details):

        if (horse['group']=="A"):
            Group_A_count+=1

        if (horse['group']=="B"):
            Group_B_count+=1

        if (horse['group']=="C"):
            Group_C_count+=1

        if (horse['group']=="D"):
            Group_D_count+=1


    # Asking and Validating The Horse Id
    while True:
        Update_horse_id = input("\nEnter Horse ID to update: ")
        if  Update_horse_id.isdigit():
                break
        else:
            print("\nInvalid input. Age must be an integer.\n")


    for horse in (total_horse_details):

        # Checking is there any horse With That Horse Id
        if horse['horse_id'] == Update_horse_id:

            print("Horse Deatils : ",horse)                    # User can View the Horse Deatails

            # Checking Which Group is Entered Horse Before Updating
            # This Part will Allow  User to Change the Group
            if (horse['group']=="A") :
                Group_A_count-=1
            elif (horse['group']=="B"):
                Group_B_count-=1
            elif (horse['group']=="C"):
                Group_C_count-=1

            elif (horse['group']=="D"):
                Group_D_count-=1
```

```
# Validating Horse Name to be an Alphabetic
while True:
    horse_name = input("\nEnter updated Horse Name: ")
    if  horse_name.isalpha():
        break
    else:
        print("\nInvalid input. Horse Name must contain only alphabetic characters.\n")

# Validating Jockey Name to be an Alphabetic
while True:
    jockey_name = input("Enter updated Jockey Name: ")
    if  jockey_name.isalpha():
        break
    else:
        print("\nInvalid input. Jockey Name must contain only alphabetic characters.\n")

#  Validating Age as Integer
while True:
    age = (input("Enter updated Age: "))
    if age.isdigit():
        break
    else:
        print("\nInvalid input. Age must be an integer.\n")

# Validating Breed Name as Alphabetic
while True:
    breed = (input("Enter updated Breed: "))
    if  breed.isalpha():
        break
    else:
        print("\nInvalid input. Breed must contain only alphabetic characters.\n")

# Asking for race Record
while True:
    Wins = input("Enter How Many Wins: ")
    Total_Races = input("Enter How Many Races: ")
    if  Wins.isdigit() and Total_Races.isdigit():
        if (int(Wins)<=int(Total_Races)):
            break
        else:
            print ("\nWins Can't be higher than Totel Races")
            continue
    else:
        print("\nInvalid input. Wins and Races must be an integer.\n")

race_record =f"{Wins} Wins in {Total_Races} races"
```

```
        # Asking For Group
        # user can put in same Group or Any Other Available Groups
        while True:
            group = str(input("Enter updated Group (A, B, C, D): ")).upper()

            if (group == "A"):
                if Group_A_count < 5 :
                    Group_A_count = Group_A_count + 1
                    break
                else:
                    print("\nA is full\n")

            elif (group == "B"):
                if Group_B_count < 5 :
                    Group_B_count = Group_B_count + 1
                    break
                else:
                    print("\nB is full\n ")

            elif (group == "C"):
                if Group_C_count < 5 :
                    Group_C_count = Group_C_count + 1
                    break
                else:
                    print("\nC is full\n ")

            elif (group == "D"):
                if Group_D_count < 5 :
                    Group_D_count = Group_D_count + 1
                    break
                else:
                    print("\nD is full\n ")

            else:
                print("\nWroung Group Input\n")

        # Inserting Updated  Data
        horse['horse_name'] = horse_name
        horse['jockey_name'] = jockey_name
        horse['age'] = age
        horse['breed'] = breed
        horse['race_record'] = race_record
        horse['group'] = group
        print("\nHorse details updated successfully!")

        return

    #IF the Horse id is  wrong
    print("Horse not found!")
```

3.2 Flow Chart

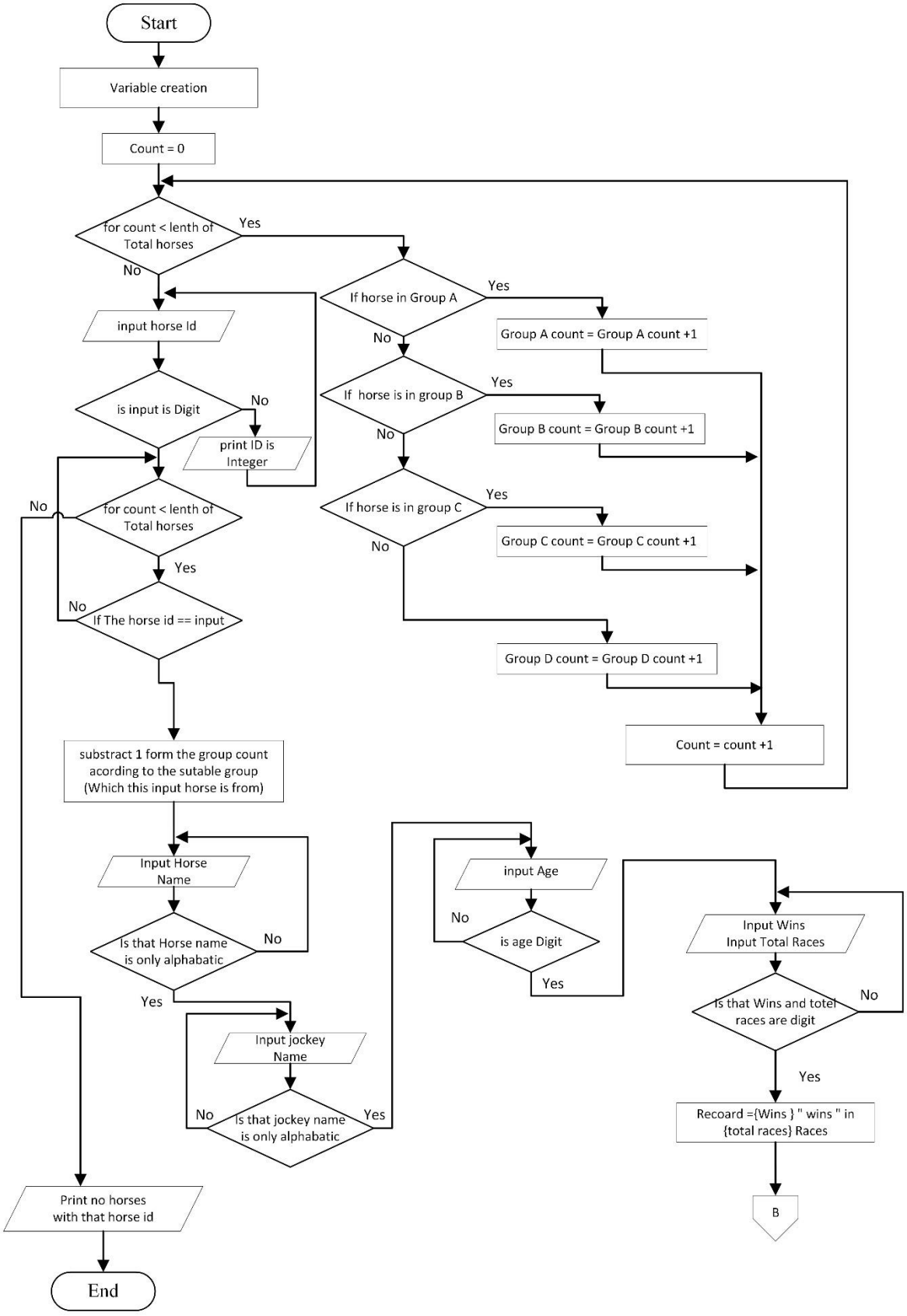

Figure 10 Flowchart 1 for UHD

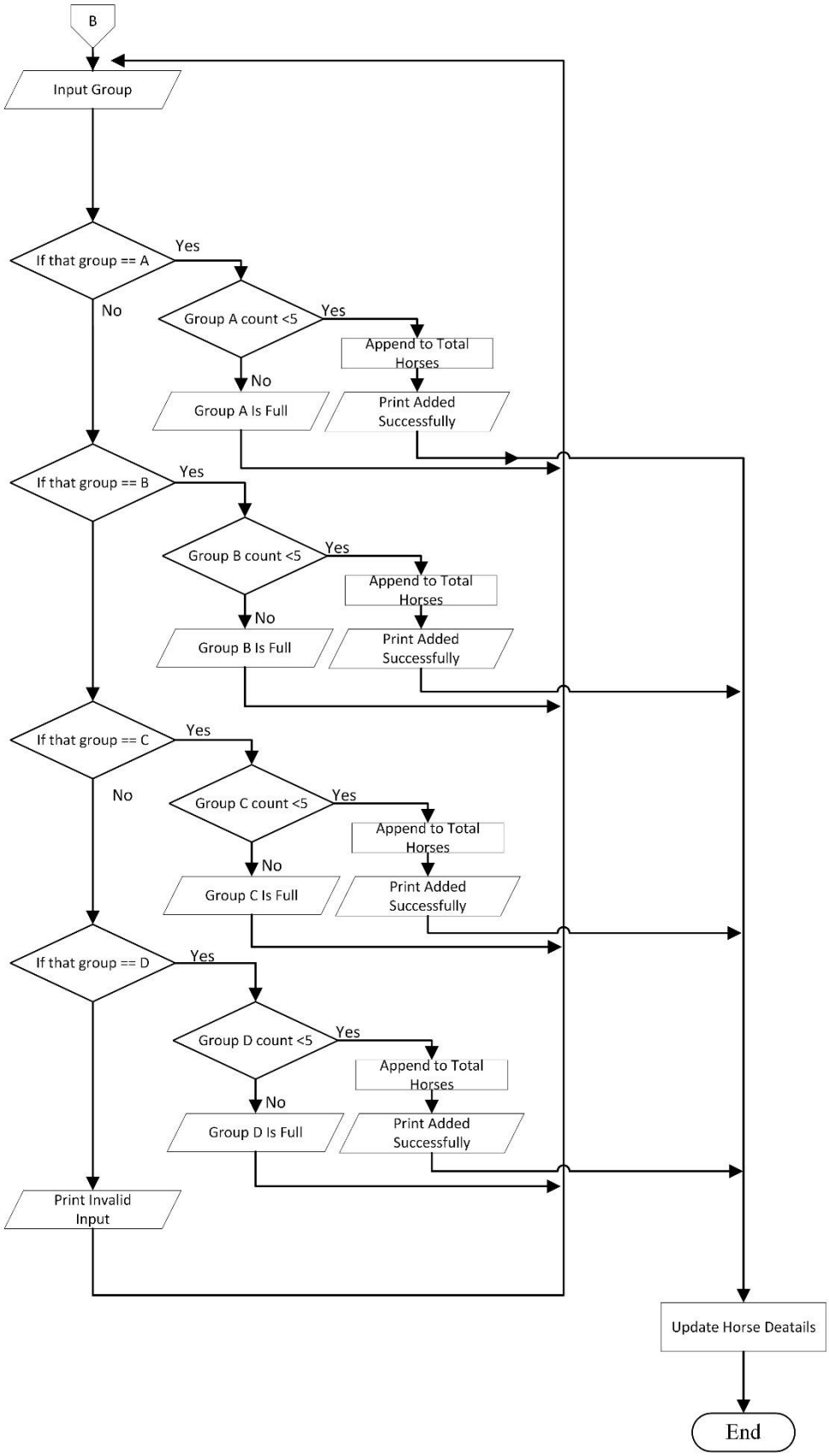

Figure 11 Flowchart 2 for UHD

3.3 Screenshot Of The Output.

```
Enter your choice: uhd

Enter Horse ID to update: 1001
Horse Deatils :  {'horse_id': '1001', 'horse_name': 'Thunderbolt', 'jockey_name': 'Alexander',
 'age': '3', 'breed': 'Arabian', 'race_record': '5 Wins in 6 races', 'group': 'A'}

Enter updated Horse Name: Thunderbolt
Enter updated Jockey Name: Alexander
Enter updated Age: 5
Enter updated Breed: Arabian
Enter How Many Wins: 5
Enter How Many Races: 6
Enter updated Group (A, B, C, D): B
Horse details updated successfully!
```

Figure 12 Updating Age And Group For horse id 1001.

```
Enter your choice: uhd

Enter Horse ID to update: 1005
Horse not found!
```

Figure 13 Tring to update Before Adding 1005

```
Enter your choice: Uhd

Enter Horse ID to update: 1004
Horse Deatils :  {'horse_id': '1004', 'horse_name': 'Cooper', 'jockey_name': 'Henry', 'age': '
5', 'breed': 'Morgan', 'race_record': '1 Wins in 5 races', 'group': 'C'}

Enter updated Horse Name: Cooper
Enter updated Jockey Name: Henry
Enter updated Age: t

Invalid input. Age must be an integer.

Enter updated Age: 5
Enter updated Breed: Moragn
Enter How Many Wins: 2
Enter How Many Races: 3
Enter updated Group (A, B, C, D): C
Horse details updated successfully!
```

Figure 14 Validation Checking

4.0 Deleting Horse Details (DHD)

At first it will ask for user to input the horse ID. Id will be an integer. If user inputs Rather than an integer it will automatically say add an integer and ask user to input

Then it will check if the horse with that horse id is in the total horse list. If not, there a message will pop up and say horse is not there.

If horse is there, then Details of the horse details will delete from the total horse .

4.1 Python Code

```
def delete_horse(total_horse_details):

    # Asking and Validating The Horse Id
    while True:
        horse_id = input("\nEnter Horse ID to delete: ")
        if  horse_id.isdigit():
            break
        else:
            print("\nInvalid input. Age must be an integer.\n")


    for horse in (total_horse_details ):

         # Checking is there any horse With That Horse Id
        if horse['horse_id'] == horse_id:

            # Removing The Horse Deatils
            total_horse_details.remove(horse)
            print("\nHorse details deleted successfully.")
            return

    print("\nHorse not found!\n")
```

4.2 Flow Chart

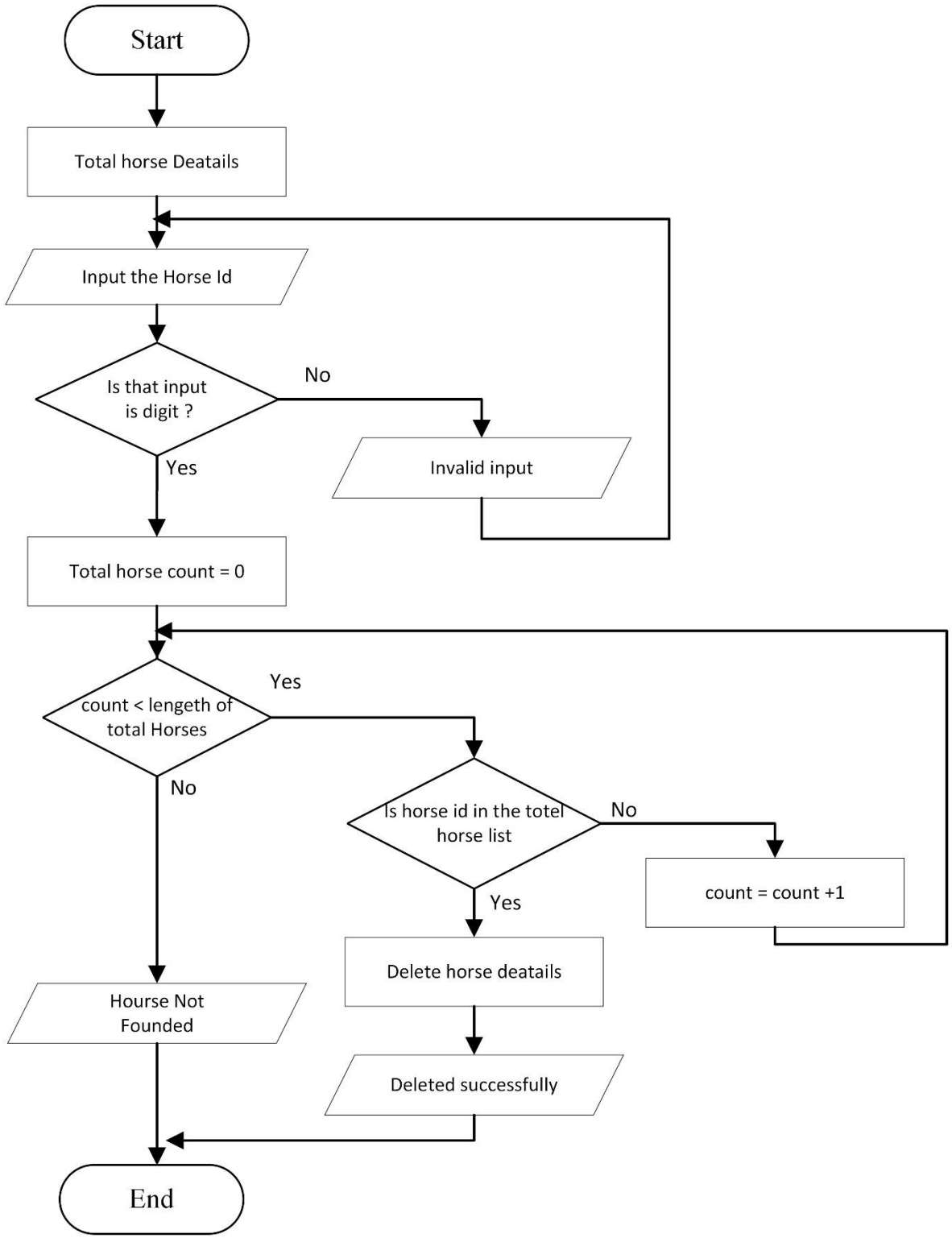

Figure 15 Flowchart for DHD

4.3 Screenshot Of The Output.

```
===== Horse Race Event Menu =====
1. Type AHD for adding horse details.
2. Type UHD for updating horse details.
3. Type DHD for deleting horse details.
4. Type VHD for viewing the registered horses' details table.
5. Type SHD for saving the horse details to the text file.
6. Type START for Start the Game

7. Type ESC to exit the program.

Enter your choice: dhd

Enter Horse ID to delete: 1004

Horse details deleted successfully.
```

Figure 16 Deleting Horse details of 1004

5.0 View Horse Details (VHD)

This function Used for viewing horse details based on order of the horse_id in assenting order.

First the outer loop iterates through each element in the list starting from the second element (index 1).

Then the current element is stored in the variable current_dict.

The variable current_horse_id is set to the integer value of the "horse_id" key in the current dictionary.

The inner loop compares the "horse_id" of the current element with the elements in the sorted part of the list, moving elements greater than the current one to the right.

The current element is then inserted into the correct position in the sorted part of the list.

Finally, the sorted list is printed.

5.1 Python Code

```
def view_horses(total_horse_details):

    for i in range(1, len(total_horse_details)):
        current_dict = total_horse_details[i]
        current_horse_id = int(current_dict["horse_id"])
        j = i - 1

        # Move elements of the sorted part that are greater than the current_horse_id
        # to one position ahead of their current position
        while j >= 0 and current_horse_id < int(total_horse_details[j]["horse_id"]):
            total_horse_details[j + 1] = total_horse_details[j]
            j -= 1

        # Insert the current_dict into the correct position in the sorted part
        total_horse_details[j + 1] = current_dict

    # Print the sorted list
    for horse in total_horse_details:
        print (horse)
```

5.2 Flow Chart

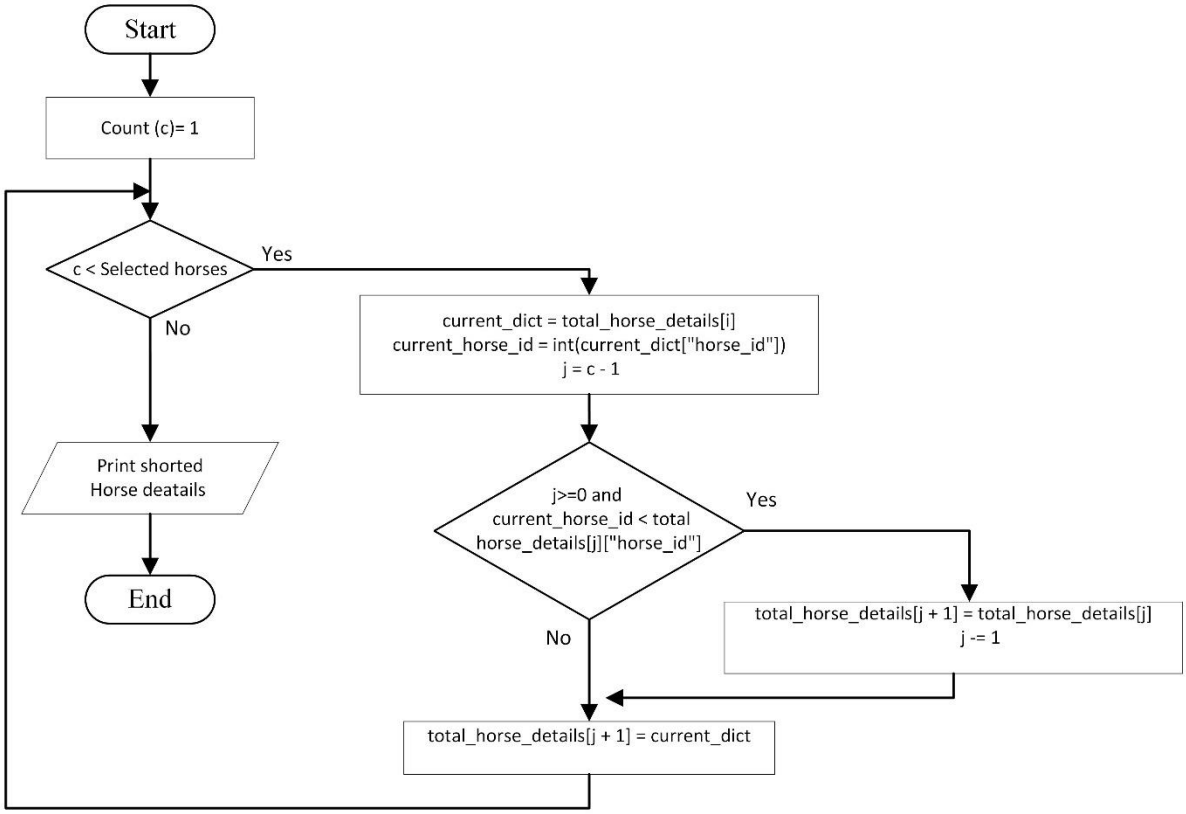

Figure 17 Flowchart for VHD

5.3 Screenshot Of The Output.

```
Enter your choice: vhd
{'horse_id': '1001', 'horse_name': 'Thunderbolt', 'jockey_name': 'Alexander', 'age': '5', 'breed': 'Arabian', 'race_record': '2 Wins in 5 races', 'group': 'A'}
{'horse_id': '1002', 'horse_name': 'Shadow', 'jockey_name': 'Benjamin', 'age': '3', 'breed': 'Mustang', 'race_record': '2 Wins in 2 races', 'group': 'B'}
{'horse_id': '1003', 'horse_name': 'Serenade', 'jockey_name': 'Christopher', 'age': '6', 'breed': 'Andalusian', 'race_record': '4 Wins in 4 races', 'group': 'C'}
{'horse_id': '1004', 'horse_name': 'Shadowfax', 'jockey_name': 'Kevin', 'age': '4', 'breed': 'Haflinger', 'race_record': '3 Wins in 5 races', 'group': 'D'}
{'horse_id': '1005', 'horse_name': 'Amber', 'jockey_name': 'Harrison', 'age': '7', 'breed': 'Haflinger', 'race_record': '0 Wins in 5 races', 'group': 'A'}
{'horse_id': '1006', 'horse_name': 'Copper', 'jockey_name': 'Patrick', 'age': '3', 'breed': 'Palomino', 'race_record': '6 Wins in 6 races', 'group': 'B'}
{'horse_id': '1007', 'horse_name': 'Sunburst', 'jockey_name': 'Oliver', 'age': '8', 'breed': 'Morgan', 'race_record': '6 Wins in 7 races', 'group': 'C'}
{'horse_id': '1008', 'horse_name': 'Moonlit', 'jockey_name': 'Liam', 'age': '4', 'breed': 'Clydesdale', 'race_record': '6 Wins in 7 races', 'group': 'D'}
{'horse_id': '1009', 'horse_name': 'Shadowfax', 'jockey_name': 'Kevin', 'age': '6', 'breed': 'Palomino', 'race_record': '2 Wins in 8 races', 'group': 'D'}
{'horse_id': '1010', 'horse_name': 'Enchanted', 'jockey_name': 'Isaac', 'age': '5', 'breed': 'Arabian', 'race_record': '2 Wins in 9 races', 'group': 'C'}
{'horse_id': '1011', 'horse_name': 'Glory', 'jockey_name': 'Patrick', 'age': '7', 'breed': 'Mustang', 'race_record': '3 Wins in 3 races', 'group': 'B'}
{'horse_id': '1012', 'horse_name': 'Ember', 'jockey_name': 'Theodore', 'age': '5', 'breed': 'Haflinger', 'race_record': '3 Wins in 4 races', 'group': 'A'}
{'horse_id': '1013', 'horse_name': 'Willow', 'jockey_name': 'Quentin', 'age': '4', 'breed': 'Quentin', 'race_record': '3 Wins in 4 races', 'group': 'A'}
{'horse_id': '1014', 'horse_name': 'Luna', 'jockey_name': 'Franklin', 'age': '3', 'breed': 'Friesian', 'race_record': '1 Wins in 9 races', 'group': 'A'}
{'horse_id': '1015', 'horse_name': 'Stormy', 'jockey_name': 'Patrick', 'age': '5', 'breed': 'Shetland', 'race_record': '6 Wins in 7 races', 'group': 'B'}
{'horse_id': '1016', 'horse_name': 'Echo', 'jockey_name': 'Nathan', 'age': '7', 'breed': 'Shetland', 'race_record': '4 Wins in 5 races', 'group': 'B'}
{'horse_id': '1017', 'horse_name': 'Thunder', 'jockey_name': 'Matthew', 'age': '6', 'breed': 'Morgan', 'race_record': '6 Wins in 8 races', 'group': 'C'}
{'horse_id': '1018', 'horse_name': 'Knight', 'jockey_name': 'Quentin', 'age': '5', 'breed': 'Thoroughbred', 'race_record': '4 Wins in 4 races', 'group': 'D'}
{'horse_id': '1019', 'horse_name': 'Sunburst', 'jockey_name': 'Franklin', 'age': '8', 'breed': 'Arabian', 'race_record': '2 Wins in 8 races', 'group': 'C'}
{'horse_id': '1020', 'horse_name': 'Charm', 'jockey_name': 'Quentin', 'age': '4', 'breed': 'Friesian', 'race_record': '2 Wins in 5 races', 'group': 'D'}
```

Figure 18 Viewing Horse details in horse id order

6.0 Save Horse Details (SHD)

SHD function is used for Saving Horse details in a text . This function has 8 variable 4 for grouping and other 4 for grouping IDs .

At first horse details in total horse will check for what group the are from then in will append to those group list and id of those horses are Group id lists.

A text file will create “horse_detail” name. That file will writeable. The group A list, group B list , group C list and group D list will be saved in that file line by line .

Then text “Group_A_id” will be create and horse id of the horses in the group A will print in the file line by line .

Then text “Group_B_id” will be create and horse id of the horses in the group B will print in the file line by line .

Then text “Group_C_id” will be create and horse id of the horses in the group C will print in the file line by line .

Then text “Group_D_id” will be create and horse id of the horses in the group D will print in the file line by line .

These id files were easy to retrieve the data.

6.1 Python Code

```
def save_to_file(total_horse_details):

    #Local Variables
    Group_A_id = [ ]
    Group_B_id = [ ]
    Group_C_id = [ ]
    Group_D_id = [ ]

    Group_A = [ ]
    Group_B = [ ]
    Group_C = [ ]
    Group_D = [ ]

    # Appending Horse Deatails and Horse IDs to Lists
    for horse in (total_horse_details):

        if (horse['group']=="A"):
            Group_A.append(horse)
            value=horse['horse_id']
            Group_A_id.append(value)

        elif (horse['group']=="B"):
            Group_B.append(horse)
            value=horse['horse_id']
            Group_B_id.append(value)

        elif (horse['group']=="C"):
            Group_C.append(horse)
            value=horse['horse_id']
            Group_C_id.append(value)

        elif (horse['group']=="D"):
            Group_D.append(horse)
            value=horse['horse_id']
            Group_D_id.append(value)

    # Saving horse deatils according to the group in one  Text file
    with open("horse_details.txt", "w") as file:

        file.write("Group A" + "\n")
        for horse in (Group_A):
            file.write(str(horse) + "\n")

        file.write("\n"+"Group B" + "\n")
        for horse in (Group_B):
            file.write(str(horse) + "\n")

        file.write("\n"+"Group C" + "\n")
        for horse in (Group_C):
            file.write(str(horse) + "\n")
```

```
    file.write("\n"+"Group D" + "\n")
    for horse in (Group_D):
        file.write(str(horse) + "\n")


# saving Horse IDs according to the group In Each Single file
with open("Group_A_id.txt","w") as file:
    for horse_id in Group_A_id:
        file.write(str(horse_id) + "\n")

with open("Group_B_id.txt","w") as file:
    for horse_id in Group_B_id:
        file.write(str(horse_id) + "\n")

with open("Group_C_id.txt","w") as file:
    for horse_id in Group_C_id:
        file.write(str(horse_id) + "\n")

with open("Group_D_id.txt","w") as file:
    for horse_id in Group_D_id:
        file.write(str(horse_id) + "\n")

print ("\nSuccesfully  Saved")
```

6.2 Flow Chart

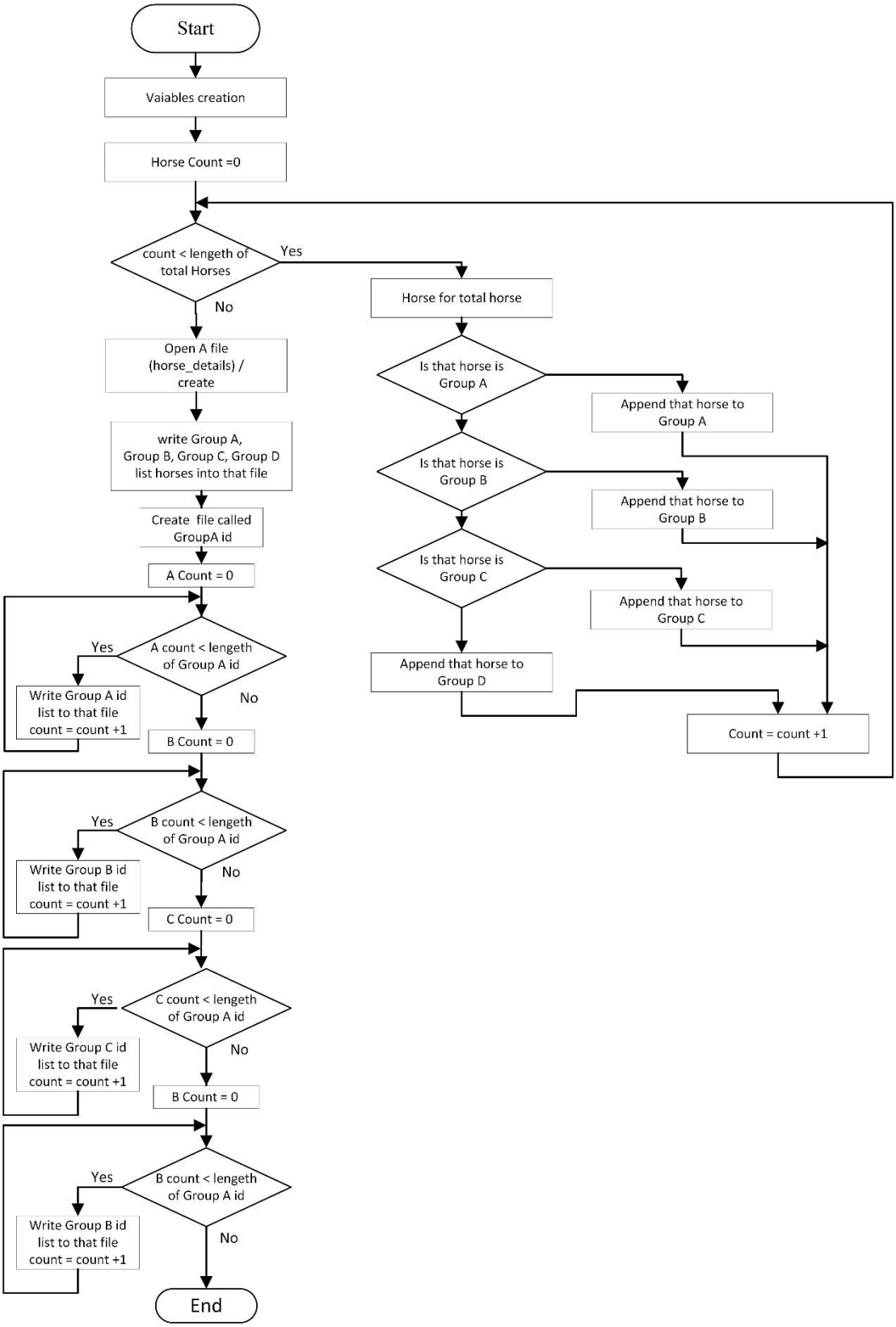

Figure 19 Flowchart for SHD

6.3 Screenshot Of The Output.

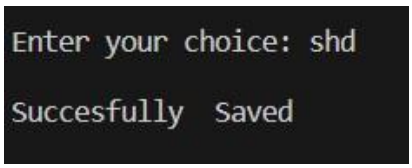

Figure 20 Saving details into Text

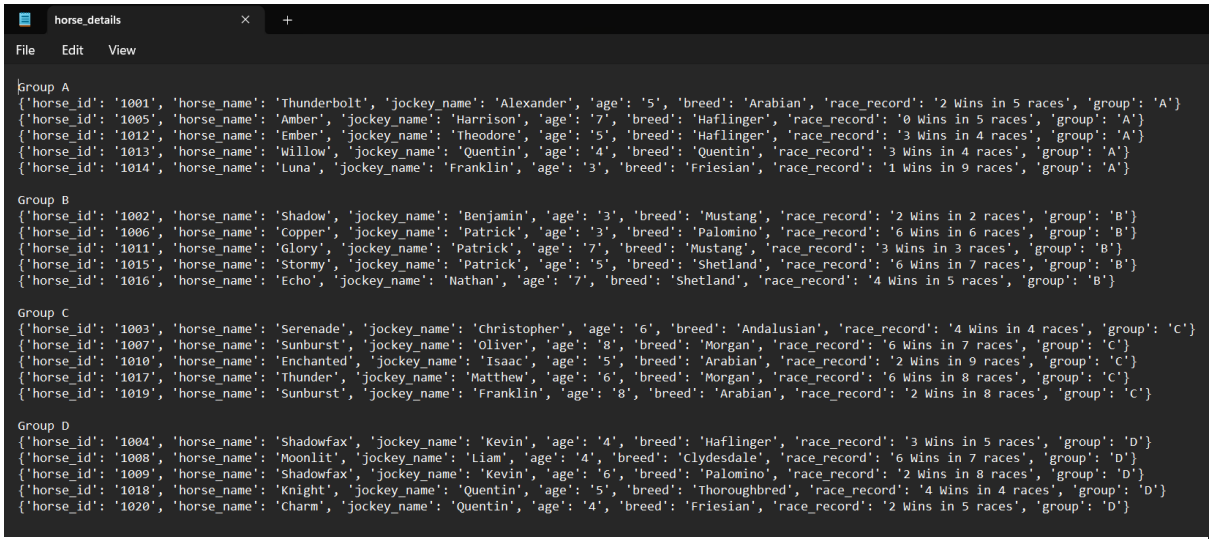

Figure 23 Txt File For Horse details

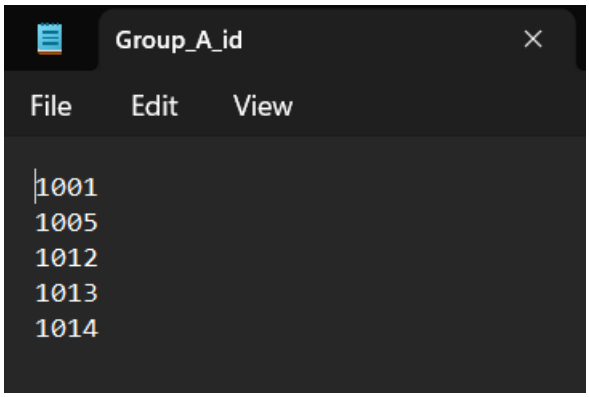

Figure 22 Group A Horse Ids in txt file

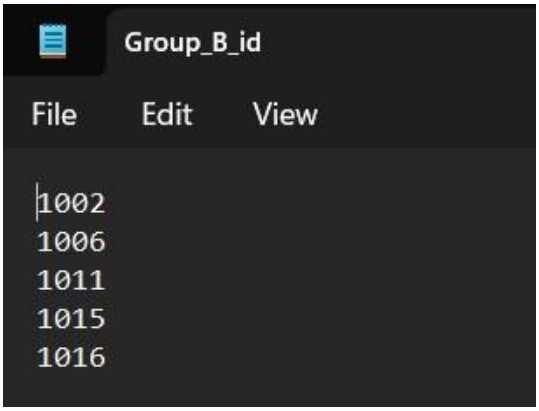

Figure 21 Group B Horse Ids in txt file

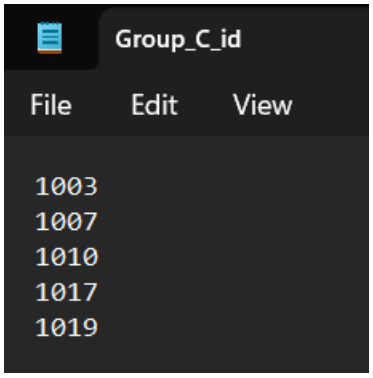

Figure 24 Group C Horse Ids in txt file

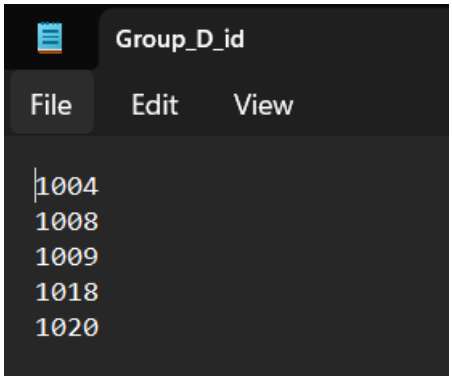

Figure 25 Group D Horse Ids in txt file

7.0 Selecting Four Horses randomly (SDD)

After saving the files we have to use this function else error message will pop up and that will tell us to do that SHD part. This function has 4 local variables to retrieve the saved data and organizing into groups.

At first this function will open “Group_A_id” in reading mode . It will read the horse ids one by one and append that to group A . same will happens to Group_B_id , Group_C_id and Group_D_id .

After that this function will randomly select an id in each group.

Then checks for those selected Ids were already in the total horse details. If not there, an error message will pop up and tell there was no horses in such horse id.

If there the Details of the horse with that horse ids will append to Selected horse details list.

7.1 Python Code

```
import random

def select_horses_for_race(total_horse_details,selected_horses):
    try:
        # Local Variable
        Group_A = []
        Group_B = []
        Group_C = []
        Group_D = []

        # Reading IDs from Saved File And appending Those into Local Groups
        with  open("Group_A_id.txt","r") as file_1:
            lines_1= file_1.readlines()
            for line in lines_1:
                removeing_enter=line.replace("\n","")
                Group_A.append(removeing_enter)

        with open("Group_B_id.txt","r") as file_2:
            lines_2= file_2.readlines()
            for line in lines_2:
                removeing_enter=line.replace("\n","")
                Group_B.append(removeing_enter)

        with open("Group_C_id.txt","r") as file_3:
            lines_3= file_3.readlines()
            for line in lines_3:
                removeing_enter=line.replace("\n","")
                Group_C.append(removeing_enter)

        with open("Group_D_id.txt","r") as file_4:
            lines_4= file_4.readlines()
            for line in lines_4:
                removeing_enter=line.replace("\n","")
                Group_D.append(removeing_enter)

        # Selecting One horse from each Group
        select_Group_A=random.choice(Group_A)
        select_Group_B=random.choice(Group_B)
        select_Group_C=random.choice(Group_C)
        select_Group_D=random.choice(Group_D)
```

```
        # Getting Deatails of  the Selected Horses and appending those deatails Selected Horses list and
Printing Thoses Deatils
        # Selected Horses is A global Variable

        for horse in total_horse_details:
            try:

                if horse['horse_id']==select_Group_A:
                    print(f"Horse id {horse['horse_id']} is selected from Group A and Horse's Name is
{horse['horse_name']}")
                    selected_horses.append(horse)

                elif horse['horse_id']==select_Group_B:
                    print(f"Horse id {horse['horse_id']} is selected from Group B and Horse's Name is
{horse['horse_name']}")
                    selected_horses.append(horse)

                elif horse['horse_id']==select_Group_C:
                    print(f"Horse id {horse['horse_id']} is selected from Group C and Horse's Name is
{horse['horse_name']}")
                    selected_horses.append(horse)

                elif horse['horse_id']==select_Group_D:
                    print(f"Horse id {horse['horse_id']} is selected from Group D and Horse's Name is
{horse['horse_name']}")
                    selected_horses.append(horse)

            except Exception:
                print("There was no horse in such Horse ID")

    except FileNotFoundError:
        print("\nFile Not Founded. Go to Menu and Save the file First")

    except Exception:
        print("\nAdd Minimum one Horse deatils for Each Group")
```

7.2 Flow Chart

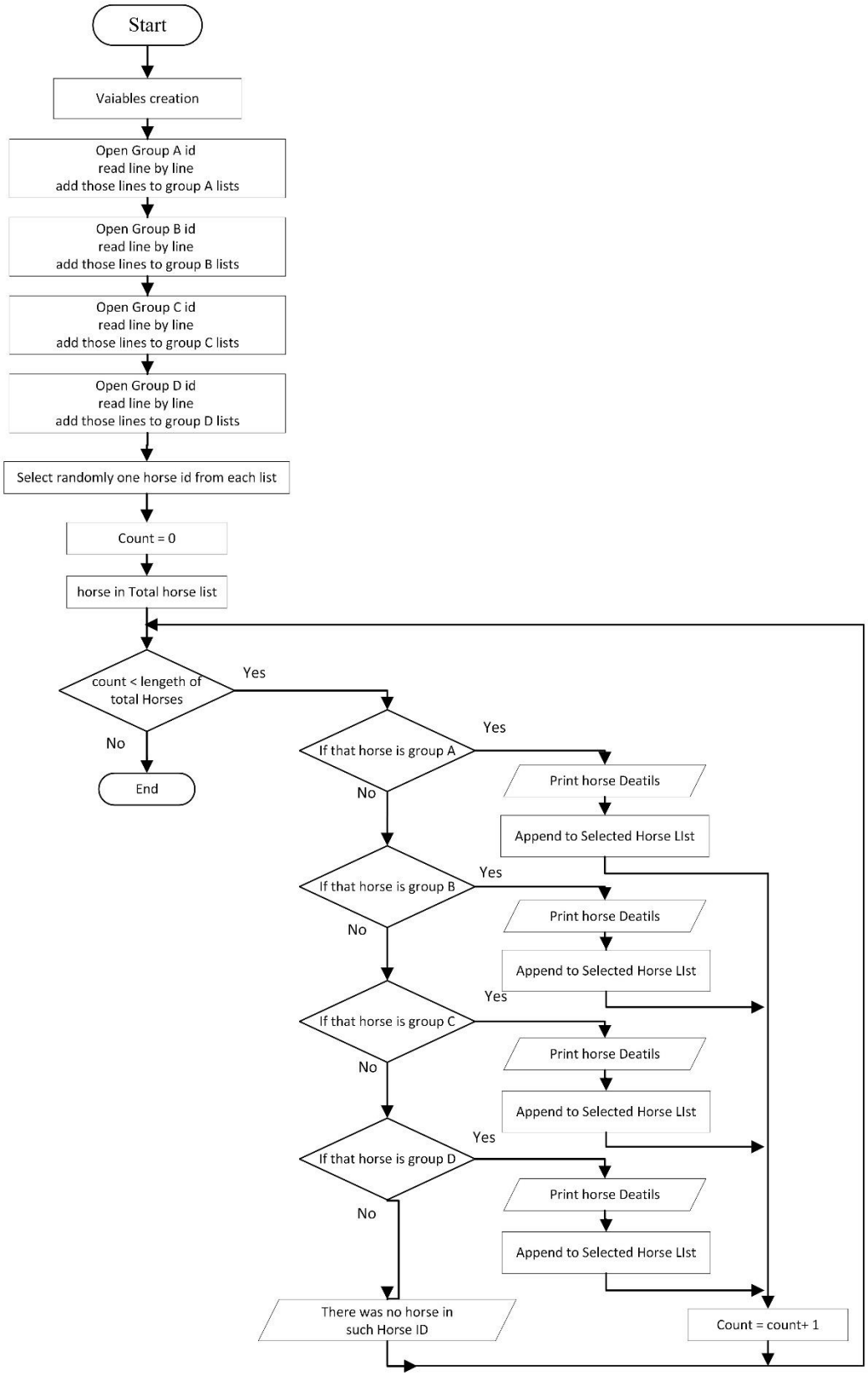

Figure 26 Flowchart for SDD

7.3 Screenshot Of The Output.

```
        ---Game is Started ---
1. Type SDD for selecting four horses randomly for the major round.
2. Type WHD for displaying the Winning horses' details.
3. Type VWH for Visualizing the time of the winning horses.
4. Type STOP  the Game And Entering the Menu .

Enter your choice: sdd
Horse id 1001 is selected from Group A and Horse's Name is Thunderbolt
Horse id 1007 is selected from Group C and Horse's Name is Sunburst
Horse id 1009 is selected from Group D and Horse's Name is Sahdowfax
Horse id 1015 is selected from Group B and Horse's Name is Stormy
```

Figure 27 Selecting Horses randomly from each group

```
Enter your choice: Start

        ---Game is Started ---
1. Type SDD for selecting four horses randomly for the major round.
2. Type WHD for displaying the Winning horses' details.
3. Type VWH for Visualizing the time of the winning horses.
4. Type STOP  the Game And Entering the Menu .

Enter your choice: sdd

Add Minimum one Horse deatils for Each Group
```

Figure 28 Trying to do SDD part before SHD part random Horses

8.0 Display Winning horses (WHD)

At first this function gives times to each selected horses randomly.

Then it will use the same exact code that use In VHD to order the horses by the timing. After that I am taking the first three horses using indexes of the list to print the horse details .

In here I assume that some horses can get same times and timing can’t be zero.

For easy purpose the Random Value will increase by 10.

8.1 Python Code

```
import random

def Winning_Horse(selected_horses):

    try:
    # Setting Random Number that increse by 10 For Each Selected Horse as time

        for Horse in selected_horses:
            Horse['time'] = random.randrange(10, 90,10)

        for i in range(1, len(selected_horses)):
            current_dict = selected_horses[i]
            current_horse_id = int(current_dict["time"])
            j = i - 1

            # Move elements of the sorted part that are greater than the current_horse_id
            # to one position ahead of their current position
            while j >= 0 and current_horse_id < int(selected_horses[j]["time"]):
                selected_horses[j + 1] = selected_horses[j]
                j -= 1

            # Insert the current_dict into the correct position in the sorted part
            selected_horses[j + 1] = current_dict


        # Printing Horse Deatils And Timing

        First_Horse=selected_horses[0]
        print (f"\nFirst Place winner Horse's ID is {First_Horse['horse_id']} \nHorse's Name is
{First_Horse['horse_name']} and Race time is {First_Horse['time']}s")

        Second_Horse=selected_horses[1]
        print (f"\nSecond Place winner Horse's ID is {Second_Horse['horse_id']} \nHorse's Name is
{Second_Horse['horse_name']} and Race time is {Second_Horse['time']}s")

        Third_Horse=selected_horses[2]
        print (f"\nThird Place winner Horse's ID is {Third_Horse['horse_id']} \nHorse's Name is
{Third_Horse['horse_name']} and Race time is {Third_Horse['time']}s")

    except Exception:
        print("\nBefore entering game menu , Finsh the Main Menu ")
        print("Input everything Order by order.")
```

8.2 Flow Chart

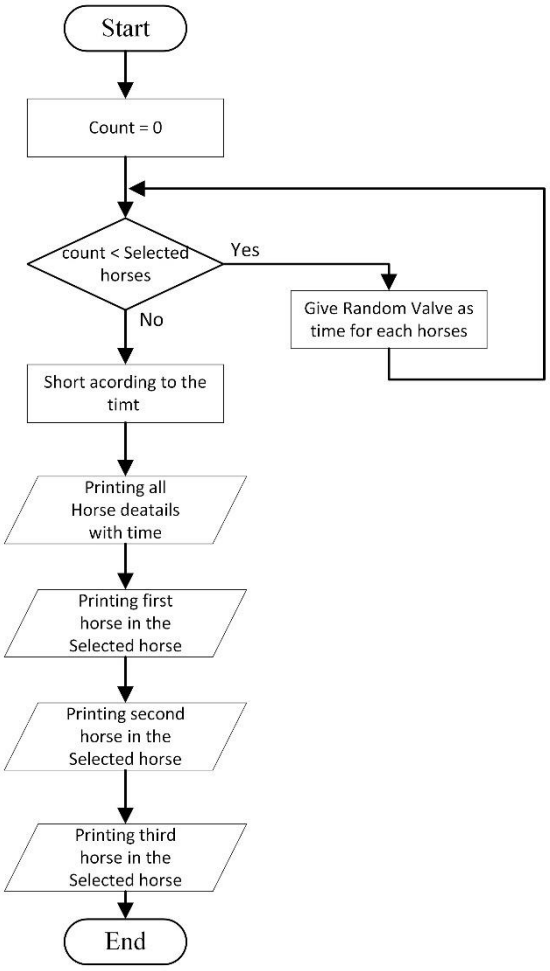

Figure 28 Flowchart for WHD

8.3 Screenshot Of The Output.

```
        ---Game is Started ---
1. Type SDD for selecting four horses randomly for the major round.
2. Type WHD for displaying the Winning horses' details.
3. Type VWH for Visualizing the time of the winning horses.
4. Type STOP  the Game And Entering the Menu .

Enter your choice: whd

First Place winner Horse's ID is 1015
Horse's Name is Stormy and Race time is 20s

Second Place winner Horse's ID is 1001
Horse's Name is Thunderbolt and Race time is 40s

Third Place winner Horse's ID is 1009
Horse's Name is Sahdowfax and Race time is 50s
```

Figure 29 Winning Hosre deatils after Time alocated

```
        ---Game is Started ---
1. Type SDD for selecting four horses randomly for the major round.
2. Type WHD for displaying the Winning horses' details.
3. Type VWH for Visualizing the time of the winning horses.
4. Type STOP  the Game And Entering the Menu .

Enter your choice: Whd

Before entering game menu , Finsh the Main Menu
Input everything Order by order.
```

Figure 30 Trying to use WHD before Completing Inputs

9.0 Visualize Winning Horses (VWH)

After selected horses' times were shorted assigning first three indexes of the selected list into variables called First_Horse, Second_Horse and Third_Horse.

Then dividing the time that assign to those horses .

After multiplying " * " by that value . It is the visualization .

Then printing that multiplied star symbol with the horse id , race time and their place

In here I assume that some horses can get same times and same count of stars

.

9.1 Python Code

```
def Visualize_Winning(selected_horses):

    try:
        # Getting First Three Horses Deatails
        First_Horse=selected_horses[0]
        Second_Horse=selected_horses[1]
        Third_Horse=selected_horses[2]

        # Checking For How may Stars
        First_time='*' * int(First_Horse['time']/10)
        Second_time='*' *int(Second_Horse['time']/10)
        Third_time='*' * int(Third_Horse['time']/10)

        # Printing Horse id , How many stars and Taken Time
        print(f"\nHorse ID {First_Horse['horse_id']} : {First_time}     {First_Horse['time']}s (1st Place)")
        print(f"Horse ID {Second_Horse['horse_id']} : {Second_time}     {Second_Horse['time']}s (2nd
Place)")
        print(f"Horse ID {Third_Horse['horse_id']} : {Third_time}     {Third_Horse['time']}s (3rd
Place)")

    except Exception:
        print("Input everything Order by order.")
```

9.2 Flow Chart

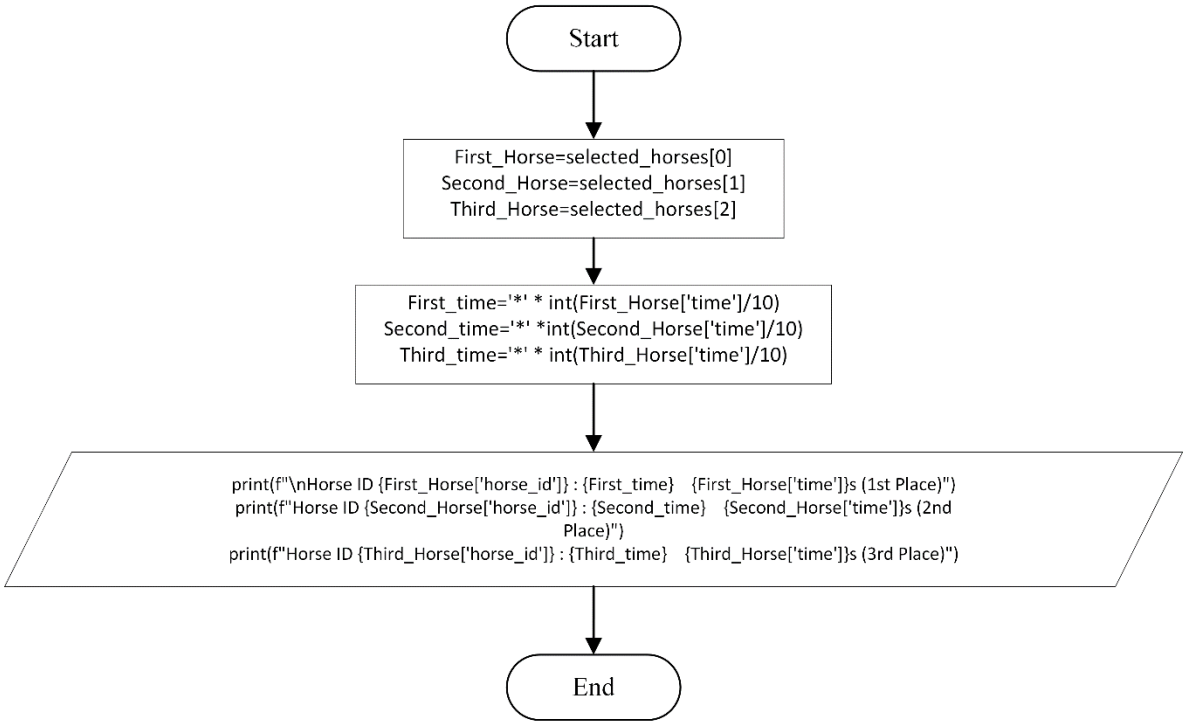

Figure 31 Flowchart for VWH

9.3 Screenshot Of The Output.

```
        ---Game is Started ---
1. Type SDD for selecting four horses randomly for the major round.
2. Type WHD for displaying the Winning horses' details.
3. Type VWH for Visualizing the time of the winning horses.
4. Type STOP  the Game And Entering the Menu .

Enter your choice: vwh

Horse ID 1015 : **    20s (1st Place)
Horse ID 1001 : ****    40s (2nd Place)
Horse ID 1009 : *****    50s (3rd Place)
```

Figure 32 Output for VWH

```
        ---Game is Started ---
1. Type SDD for selecting four horses randomly for the major round.
2. Type WHD for displaying the Winning horses' details.
3. Type VWH for Visualizing the time of the winning horses.
4. Type STOP  the Game And Entering the Menu .

Enter your choice: vwh
Input everything Order by order.
```

Figure 33 trying to Do by unorder