BSc. Artificial Intelligence & Data Science

Level 04

CM 1601

Programming Fundamentals

COURSEWORK

Rapid Run

ROSHANA CHARLES

IIT ID: 20232078

RGU STUDENT ID: 2331410

Table of Contents:

List of Figures:

Command Line Menu:

Code:

```
import random

print("Welcome to the Most anticipated event in the town: 'Rapid Run'\n")
print('''----------Operations----------\n
    AHD   Add Horse Details
    UHD   Update Horse Details
    DHD   Delete Horse Details
    VHD   View Horse Details
    SHD   Save Horse Details
    SDD   Select Four Horses Randomly
    WHD   Winning Horses' Details
    VWH   Visualize Winning Horse
    ESC   Exit''')
```

Description:

- 'Command Line Menu' provides an interface to interact with the program and helps the user to select the functions he wants to execute.

Output:

```
Welcome to the Most anticipated event in the town: 'Rapid Run'

----------Operations----------

    AHD   Add Horse Details
    UHD   Update Horse Details
    DHD   Delete Horse Details
    VHD   View Horse Details
    SHD   Save Horse Details
    SDD   Select Four Horses Randomly
    WHD   Winning Horses' Details
    VWH   Visualize Winning Horse
    ESC   Exit
Horse details loaded successfully.
What would you like to do?
```

Figure 1 – Command Line Menu Output

Loading Horse Details:

Code:

```
horse_details=[]

def load_horse_details_from_file(file_path):
    loaded_horse_details = []
    try:
        with open(file_path, 'r') as file:
            for line in file:
                fields = line.strip().split()
                horse = {
                    "Horse ID": fields[0],
                    "Horse Name": fields[1],
                    "Jockey Name": fields[2],
                    "Age": fields[3],
                    "Breed": fields[4],
                    "Race Record": fields[5],
                    "Group": fields[6]
                }
                loaded_horse_details.append(horse)
        print("Horse details loaded successfully.")
    except FileNotFoundError:
        print("File not found. No horse details loaded.")
    except Exception as e:
        print(f"An error occurred while loading horse details: {e}")

    return loaded_horse_details

file_path = "D:\\Users\\WIN XI\\PycharmProjects\\pythonProject1\\horse_details.txt"
horse_details = load_horse_details_from_file(file_path)
```

Description:

- Empty horse_details[] list is created to store details of the horses when the user inserts them.
- load_horse_details_from_file() function loads all the data that is stored in horse_details file. It helps other functions to retrieve and edit data easily when they are executed.

Output:

```
Horse details loaded successfully.
```

Figure 2 – Loading Horse Details Output

AHD – Add Horse Details:

Code:

```
def add_horse_details():
    horse_id=input("Enter Horse ID: ")
    horse_name=input("Enter Horse Name: ")
    jockey_name=input("Enter Jockey Name: ")
    age=input("Enter Age: ")
    breed=input("Enter Breed: ")
    race_record=input("Enter Race Record: ")
    group=input("Enter group: ")

    horse_details.append({"Horse ID": horse_id, "Horse Name": horse_name, "Jockey
                          Name": jockey_name, "Age": age, "Breed": breed, "Race Record":
                          race_record, "Group": group})
```

Description:

- add_horse_details() function takes horse details from the user and stores them in a dictionary.
- Input function directs the user to insert data accordingly.
- horse_details.append() function adds the newly inserted data to the horse_details list created before.

Output:

```
What would you like to do? AHD
Enter Horse ID: 1013
Enter Horse Name: Flash
Enter Jockey Name: Wallace
Enter Age: 4yrs
Enter Breed: Standardbred
Enter Race Record: 6/8
Enter group: D
```

Figure 3 – AHD Output

Flowchart:

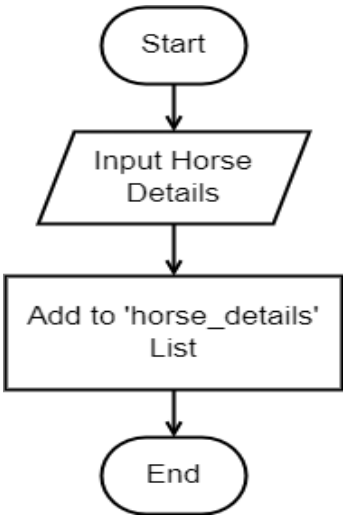

Figure 4 – AHD Flowchart

SHD – Save Horse Details:

Code:

```
def save_horse_details():

    def sort_horse_details(horse_details):
        n = len(horse_details)
        for i in range(n - 1):
            for j in range(0, n - i - 1):
                if horse_details[j]['Group'] > horse_details[j + 1]['Group']:
                    temp = horse_details[j]
                    horse_details[j] = horse_details[j + 1]
                    horse_details[j + 1] = temp

    sort_horse_details(horse_details)

    f = open('horse_details.txt', 'w')
    for horse in horse_details:
        f.write(f"{horse['Horse ID']}  {horse['Horse Name']}  {horse['Jockey Name']}
                {horse['Age']}  {horse['Breed']}  {horse['Race Record']}
                {horse['Group']}\n")
    f.close()
    print("Horse Details Saved to horse_details.txt Successfully!")
```

Description:

- sort_horse_details() function sorts the horses according to the ascending order of their group.
- f = open() function opens the horse_details.txt file and writes the inserted data.
- This function can also be called inside another function, so that the user doesn't have to execute SHD manually every time a change is made.

Output:

```
What would you like to do? SHD
Horse Details Saved to horse_details.txt Successfully!
```

Figure 5 – SHD Output

horse_details

File Edit View

```
1008  Sapphire  Morgan  6yrs  Thoroughbred  7/10  A
1005  Buk  Charles  5yrs  Appaloosa  5/6  A
1004  Meadow  Robert  8yrs  Appaloosa  2/5  A
1007  Sheriff  John  7yrs  Standardbred  5/6  B
1006  Blaze  Andrew  4yrs  Thoroughbred  4/6  B
1015  Cisco  Scott  8yrs  Thoroughbred  10/11  B
1003  Thunderbolt  Peter  5yrs  Standardbred  4/5  C
1001  Bella  Bob  5yrs  Standardbred  4/10  C
1012  Chief  Matthew  6yrs  Appaloosa  6/9  C
1002  Shadowfax  Jordan  6yrs  Thoroughbred  8/11  D
1009  Barley  Strobel  5yrs  Standardbred  3/9  D
1011  Spirit  Thomas  7yrs  Appaloosa  9/9  D
1013  Flash  Wallace  4yrs  Standardbred  6/8  D
```

Figure 6 – SHD Output in horse_details.txt

Flowchart:

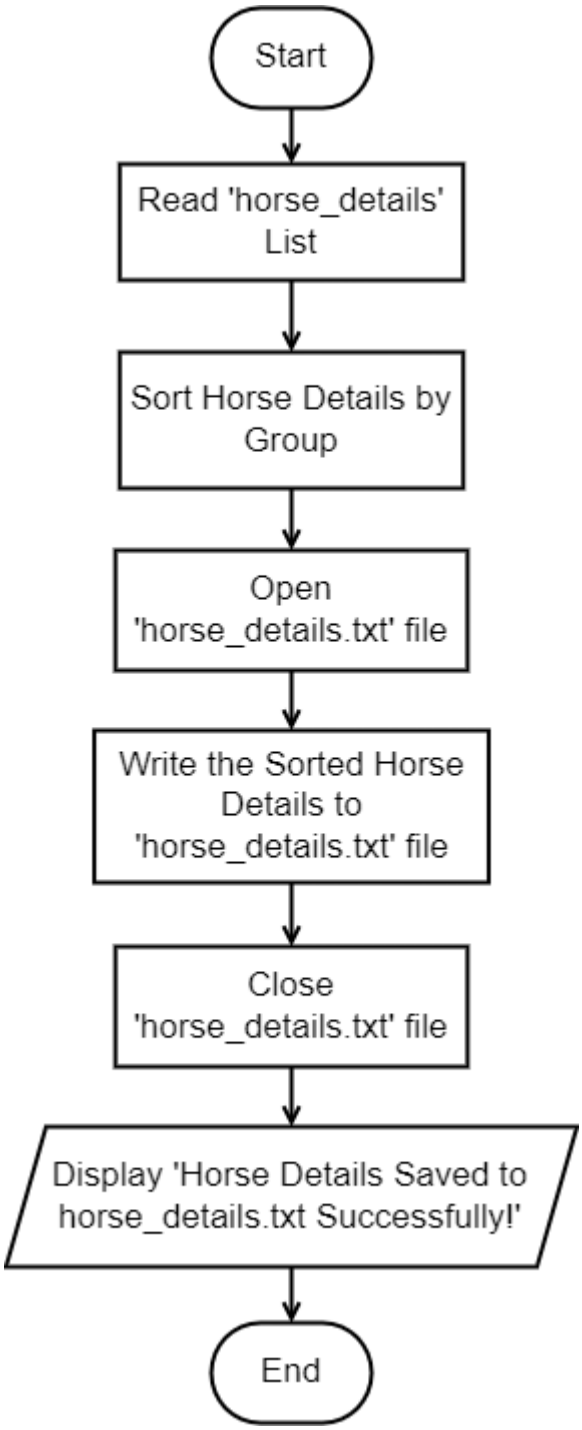

Figure 7 – SHD Flowchart

UHD – Update Horse Details:

Code:

```
def update_horse_details():
    horse_id = input("Enter the ID of the Horse to Update: ")

    global horse_details

    for horse in horse_details:
        if horse["Horse ID"] == horse_id:
            horse["Horse Name"] = input("Enter Updated Horse Name: ")
            horse["Jockey Name"] = input("Enter Updated Jockey Name: ")
            horse["Age"] = input("Enter Updated Age: ")
            horse["Breed"] = input("Enter Updated Breed: ")
            horse["Race Record"] = input("Enter Updated Race Record: ")
            horse["Group"] = input("Enter Updated Group: ")
            print("Horse Details Updated Successfully!")
            save_horse_details()
            return

    print("Horse does not Exist.")
```

Description:

- When the user inserts the horse_id of the horse he wants to update, for loop searches for it in horse_details[] list.
- If the entered horse id exists, the program updates the given details by assigning those values according to the 'key' of the dictionary.
- If the entered horse id does not exist, the program displays 'Horse does not Exist.' message.
- SHD function is called inside this function to automatically save the newly updated data.

Output:

```
What would you like to do? UHD
Enter the ID of the Horse to Update: 1013
Enter Updated Horse Name: Flash
Enter Updated Jockey Name: Jim
Enter Updated Age: 4yrs
Enter Updated Breed: Standardbred
Enter Updated Race Record: 8/10
Enter Updated Group: D
Horse Details Updated Successfully!
Horse Details Saved to horse_details.txt Successfully!
```

Figure 8 – UHD Output

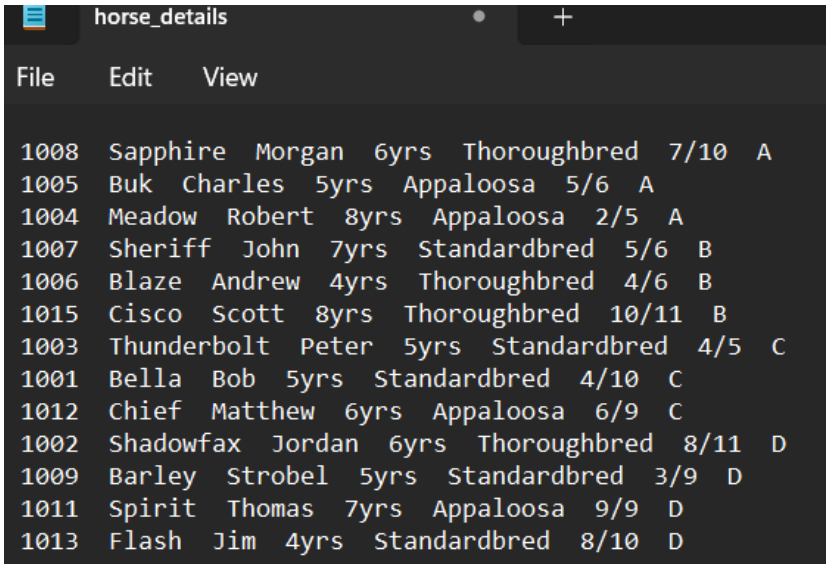

Figure 9 – UHD Output in horse_details.txt file

Flowchart:

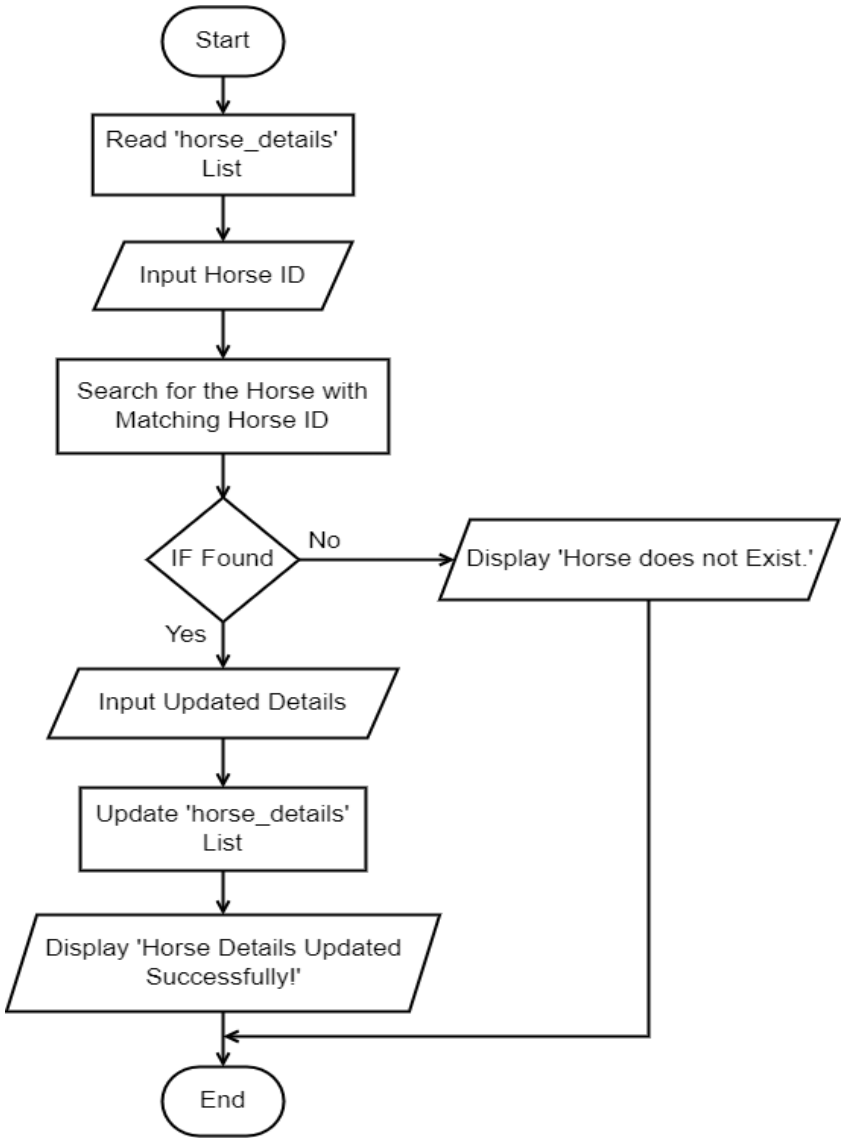

Figure 10 – UHD Flowchart

DHD – Delete Horse Details:

Code:

```
def delete_horse_details():
    horse_id = input("Enter the ID of the Horse to Delete: ")

    global horse_details

    for horse in horse_details:
        if horse["Horse ID"] == horse_id:
            horse_details.remove(horse)
            print("Horse Details Deleted Successfully!")
            save_horse_details()
            return

    print("Horse does not Exist.")
```

Description:

- When the user inserts the horse_id of the horse he wants to delete, for loop searches for it in horse_details[] list.
- If the entered horse id exists, the program deletes all the data related to it.
- If the entered horse id does not exist, the program displays 'Horse does not Exist.' message.
- SHD function is called inside this function to automatically save data.

Output:

```
What would you like to do? DHD
Enter the ID of the Horse to Delete: 1013
Horse Details Deleted Successfully!
Horse Details Saved to horse_details.txt Successfully!
```

Figure 11 – DHD Output

```
horse_details
File    Edit    View

1008  Sapphire  Morgan  6yrs  Thoroughbred  7/10  A
1005  Buk  Charles  5yrs  Appaloosa  5/6  A
1004  Meadow  Robert  8yrs  Appaloosa  2/5  A
1007  Sheriff  John  7yrs  Standardbred  5/6  B
1006  Blaze  Andrew  4yrs  Thoroughbred  4/6  B
1015  Cisco  Scott  8yrs  Thoroughbred  10/11  B
1003  Thunderbolt  Peter  5yrs  Standardbred  4/5  C
1001  Bella  Bob  5yrs  Standardbred  4/10  C
1012  Chief  Matthew  6yrs  Appaloosa  6/9  C
1002  Shadowfax  Jordan  6yrs  Thoroughbred  8/11  D
1009  Barley  Strobel  5yrs  Standardbred  3/9  D
1011  Spirit  Thomas  7yrs  Appaloosa  9/9  D
```

Figure 12 - DHD Output in horse_details.txt file

Flowchart:

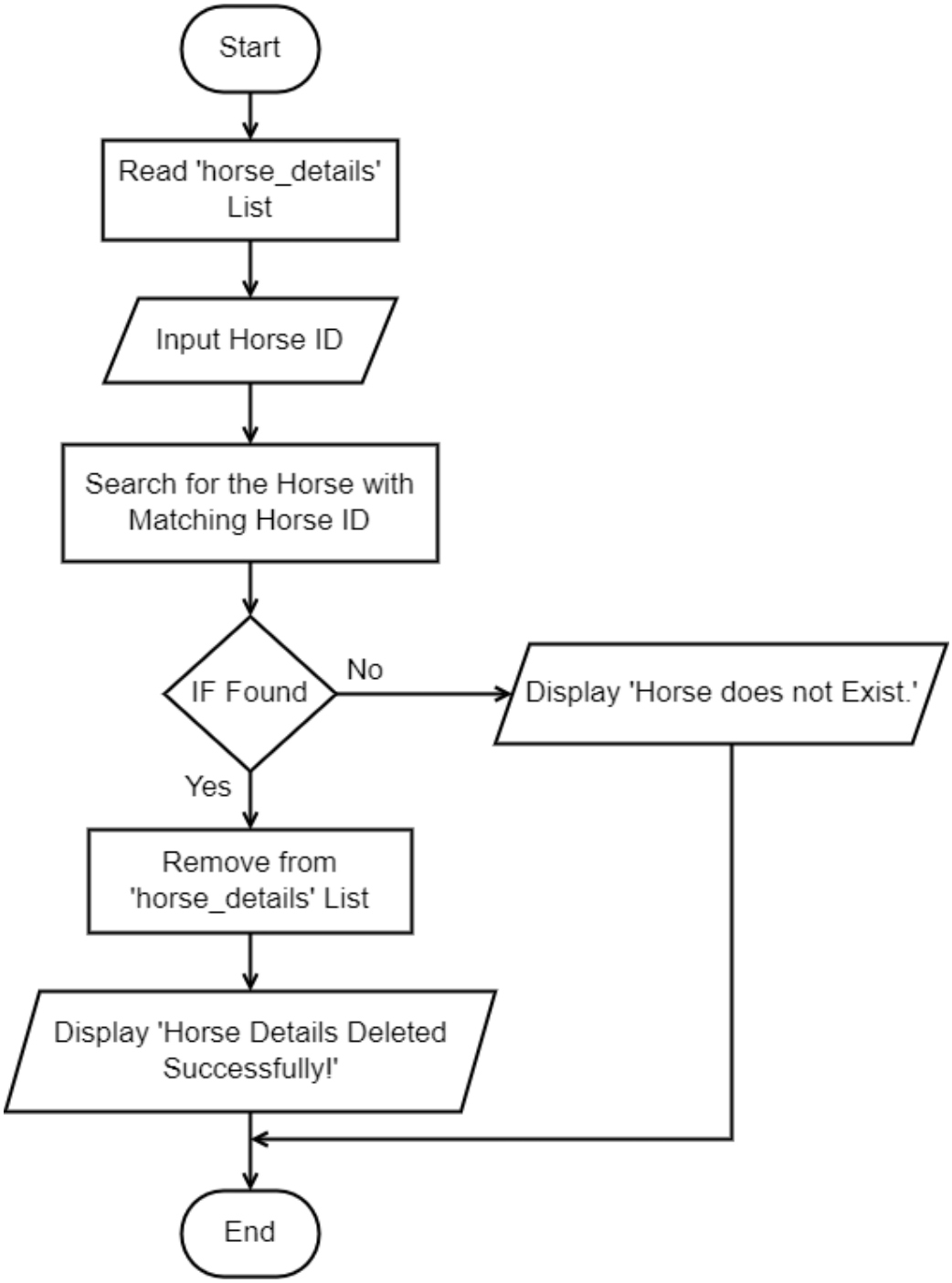

Figure 13 – DHD Flowchart

VHD – View Horse Details:

Code:

```
def view_horse_details():

    def sort_horse_details(horse_details):
        n = len(horse_details)
        for i in range(n - 1):
            for j in range(0, n - i - 1):
                if horse_details[j]['Horse ID'] > horse_details[j + 1]['Horse ID']:
                    temp = horse_details[j]
                    horse_details[j] = horse_details[j + 1]
                    horse_details[j + 1] = temp

    sort_horse_details(horse_details)

    print("{:<10} {:<20} {:<20} {:<10} {:<20} {:<15} {:<10}".format("Horse ID",
          "Horse Name", "Jockey Name", "Age", "Breed", "Race Record", "Group"))
    print("-" * 105)

    for horse in horse_details:
        print("{:<10} {:<20} {:<20} {:<10} {:<20} {:<15} {:<10}".format(horse['Horse ID'],
                horse['Horse Name'], horse['Jockey Name'], horse['Age'], horse['Breed'],
                horse['Race Record'], horse['Group']))
```

Description:

- sort_horse_details() function sorts the horses by their horse id.
- Horse details are displayed in columns.

Output:

```
What would you like to do? VHD
Horse ID   Horse Name          Jockey Name         Age        Breed               Race Record     Group
---------------------------------------------------------------------------------------------------------
1001       Bella               Bob                 5yrs       Standardbred        4/10            C
1002       Shadowfax           Jordan              6yrs       Thoroughbred        8/11            D
1003       Thunderbolt         Peter               5yrs       Standardbred        4/5             C
1004       Meadow              Robert              8yrs       Appaloosa           2/5             A
1005       Buk                 Charles             5yrs       Appaloosa           5/6             A
1006       Blaze               Andrew              4yrs       Thoroughbred        4/6             B
1007       Sheriff             John                7yrs       Standardbred        5/6             B
1008       Sapphire            Morgan              6yrs       Thoroughbred        7/10            A
1009       Barley              Strobel             5yrs       Standardbred        3/9             D
1011       Spirit              Thomas              7yrs       Appaloosa           9/9             D
1012       Chief               Matthew             6yrs       Appaloosa           6/9             C
1015       Cisco               Scott               8yrs       Thoroughbred        10/11           B
```

Figure 14 – VHD Output

Flowchart:

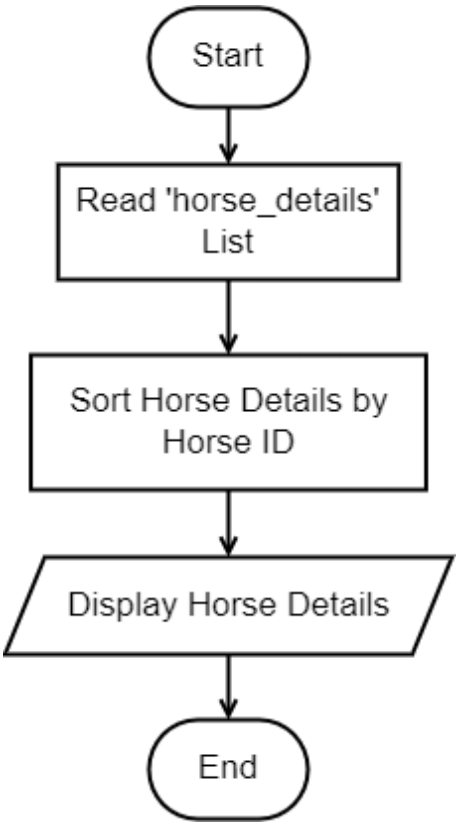

Figure 15 – VHD Flowchart

SDD – Select Four Horses Randomly:

Code:

```
selected_horses=[]

def select_horses():
    try:
        unique_group = set(horse['Group'] for horse in horse_details)

        for group in unique_group:
            group_horse = [horse for horse in horse_details if horse['Group'] == group]

            try:
                selected_horse = random.choice(group_horse)
                selected_horses.append(selected_horse)
            except IndexError:
                print(f"Error: No horses found in group {group}. Skipping this group.")

        print("Horses are Selected")
        print(selected_horses)
    except Exception as e:
        print(f"An error occurred: {e}")
```

Description:

- selected_horses[] list is created outside the function, because it is accessed by other functions too.
- This function randomly selects 4 horses from each group, stores them in selected_horses[] list and displays their details.

Output:

```
What would you like to do? SDD
Horses are Selected
[{'Horse ID': '1003', 'Horse Name': 'Thunderbolt', 'Jockey Name': 'Peter',
 'Age': '5yrs', 'Breed': 'Standardbred', 'Race Record': '4/5', 'Group': 'C'},
 {'Horse ID': '1008', 'Horse Name': 'Sapphire', 'Jockey Name': 'Morgan', 'Age':
  '6yrs', 'Breed': 'Thoroughbred', 'Race Record': '7/10', 'Group': 'A'},
 {'Horse ID': '1002', 'Horse Name': 'Shadowfax', 'Jockey Name': 'Jordan',
 'Age': '6yrs', 'Breed': 'Thoroughbred', 'Race Record': '8/11', 'Group': 'D'},
 {'Horse ID': '1015', 'Horse Name': 'Cisco', 'Jockey Name': 'Scott', 'Age':
 '8yrs', 'Breed': 'Thoroughbred', 'Race Record': '10/11', 'Group': 'B'}]
```

Figure 16 – SDD Output

Flowchart:

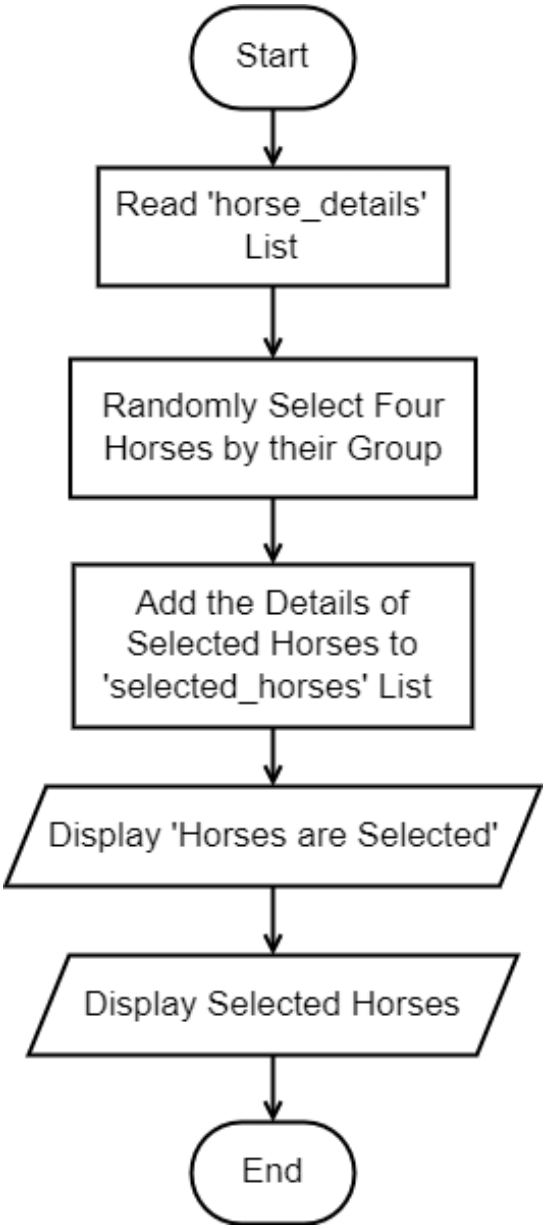

Figure 17 – SDD Flowchart

WHD – Winning Horses' Details:

Code:

```
def display_winning_horses():

    import random
    def assign_random_times(selected_horses):
        for horse in selected_horses:
            horse['Race Time'] = random.randint(0, 90)

    def sort_horses_by_race_time(selected_horses):
        n = len(selected_horses)
        for i in range(n - 1):
            for j in range(0, n - i - 1):
                if selected_horses[j]['Race Time'] > selected_horses[j + 1]['Race Time']:
                    selected_horses[j], selected_horses[j + 1] = selected_horses[j + 1],
                    selected_horses[j]

    assign_random_times(selected_horses)
    sort_horses_by_race_time(selected_horses)

    print("\nFinal Positions:")
    print("{:<10} {:<20} {:<10}".format("Horse ID", "Horse Name", "Race Time"))
    print("-" * 40)

    positions = ["1st", "2nd", "3rd"]
    i = 0
    while i < len(positions):
        horse = selected_horses[i]
        print("{:<10} {:<20} {:<10}".format(horse['Horse ID'], horse['Horse Name'],
              horse['Race Time']))
        i += 1
```

Description:

- assign_random_times() function assigns a random time for the four selected horses.
- sort_horses_by_race_time() function sorts the selected horses by their race time.
- Then selects the 1st, 2nd and 3rd places and displays them excluding the last place.

Output:

```
What would you like to do? WHD

Final Positions:
Horse ID   Horse Name           Race Time
----------------------------------------
1008       Sapphire             9
1015       Cisco                45
1003       Thunderbolt          58
```

Figure 18 – WHD Output

Flowchart:

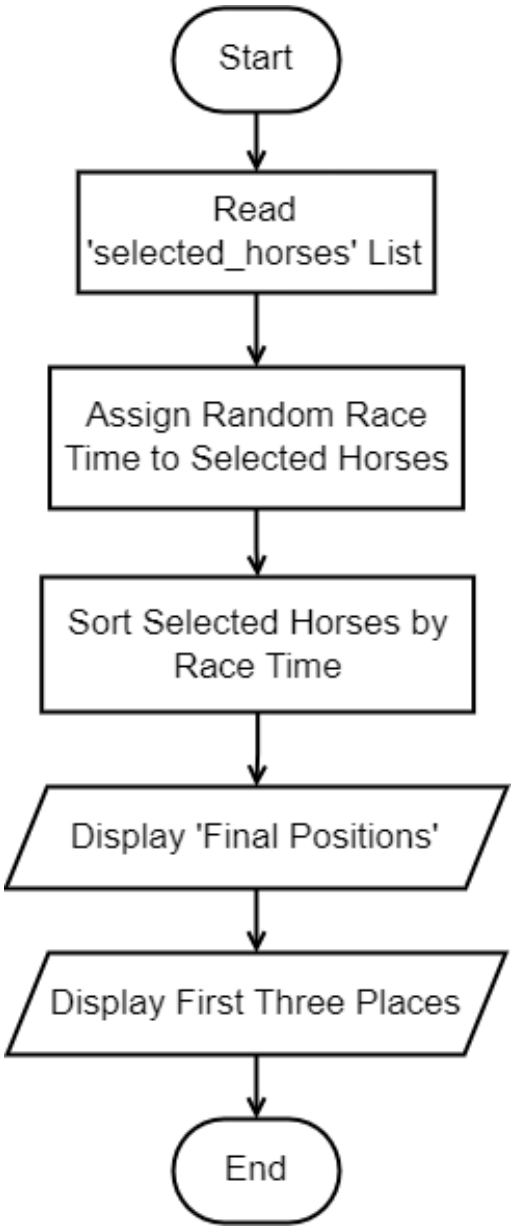

Figure 19 – WHD Flowchart

VWH – Visualize Winning Horses:

Code:

```
def visualize_winning_horses():
    def get_position_text(horse):
        race_times = [h['Race Time'] for h in selected_horses]

        min_race_time = min(race_times)
        second_min_race_time = min(val for val in race_times if val != min_race_time)
        third_min_race_time = min(val for val in race_times if val != min_race_time and
                                  val != second_min_race_time)

        if horse['Race Time'] == min_race_time:
            return "1st"
        elif horse['Race Time'] == second_min_race_time:
            return "2nd"
        elif horse['Race Time'] == third_min_race_time:
            return "3rd"

    def visualize_race_time(horse):
        position = get_position_text(horse)
        if position:
            bar_length = horse['Race Time'] // 10
            stars = '*' * bar_length
            print(f"{horse['Horse Name']}: {stars} {horse['Race Time']}s ({position}
                  Place)")

    for horse in selected_horses:
        visualize_race_time(horse)
```

Description:

- get_position_text() function assigns positions to winning horses according to their race time.
- visualize_race_time() function visualizes the time of the first three places.

Output:

```
What would you like to do? VWH
Sapphire:  9s (1st Place)
Cisco: **** 45s (2nd Place)
Thunderbolt: ***** 58s (3rd Place)
```

Figure 20 – VWH Output

Flowchart:

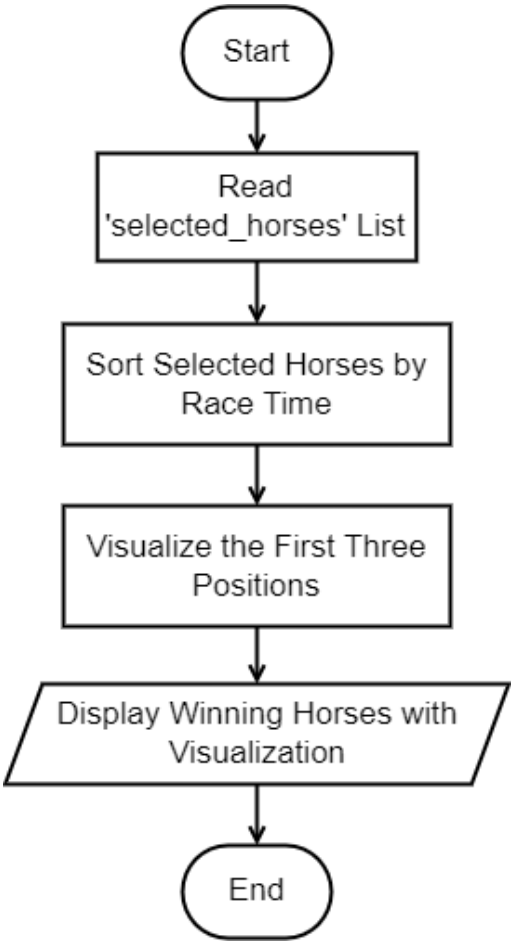

Figure 21 – VWH Flowchart

Executing/Calling the Functions:

Code:

```
while True:
    user_input=input("What would you like to do? ").upper()
    if user_input=='AHD':
        add_horse_details()
    elif user_input=='UHD':
        update_horse_details()
    elif user_input=='DHD':
        delete_horse_details()
    elif user_input=='VHD':
        view_horse_details()
    elif user_input=='SHD':
        save_horse_details()
    elif user_input=='SDD':
        select_horses()
    elif user_input=='WHD':
        display_winning_horses()
    elif user_input=='VWH':
        visualize_winning_horses()
    elif user_input=='ESC':
        print("Exiting...Thank You for Your Cooperation!")
        break
    else:
        print("Invalid Action. Please Try Again.")
```

Description:

- while True function executes the relevant function according to the user input.
- It terminates the program when user input is 'ESC'.
- It displays 'Invalid Action. Please Try Again.' message when the user inputs a wrong code.

Output:

```
What would you like to do?
```

Figure 22 – Output for Getting the Function from the User

```
What would you like to do? ESC
Exiting...Thank You for Your Cooperation!

Process finished with exit code 0
```

Figure 23 – Output for Exiting the Program

```
What would you like to do? SDW
Invalid Action. Please Try Again.
```

Figure 24 – Output for Invalid Function

References

- *Google Books* (no date). https://www.google.lk/books/edition/Dive_Into_Python/7MmeyhTnqRkC?hl=en&gbpv=0.