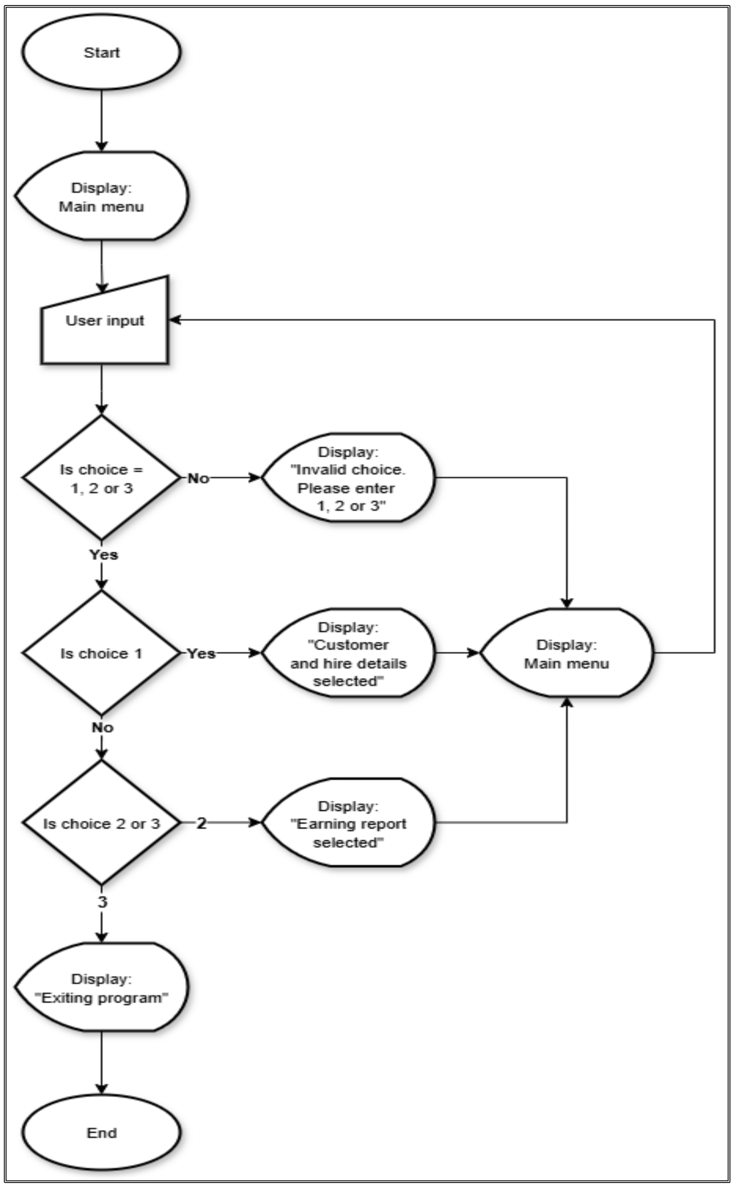
Table of Contents

Introduction

This report will outline the development process for the creation of a new software system for a small shop that loans fishing and camping equipment. Each task utilized the program development cycle where a design was created in the form of flowcharts and pseudocode, where after code was written in accordance with the design using appropriate syntax. Once all syntax errors were corrected and program was in an executable state, extensive testing was done to verify if the logic worked as intended. In the cases where results showed logic errors the process was repeated until no error was found (Gaddis, 2024).

Task 1

Flowchart



Start
Display: Main menu
User input
Is choice = 1, 2 or 3
No
Display: "Invalid choice. Please enter 1, 2 or 3"
Yes
Is choice 1
Yes
Display: "Customer and hire details selected"
Display: Main menu
No
Is choice 2 or 3
2
Display: "Earning report selected"
3
Display: "Exiting program"
End

Figure 1 - Task 1 Flowchart

The flowchart as seen in figure 1 represents the algorithm used for the main menu. When program starts, the users is presented with three options (Customer and hire details, Earnings report, and Exit).

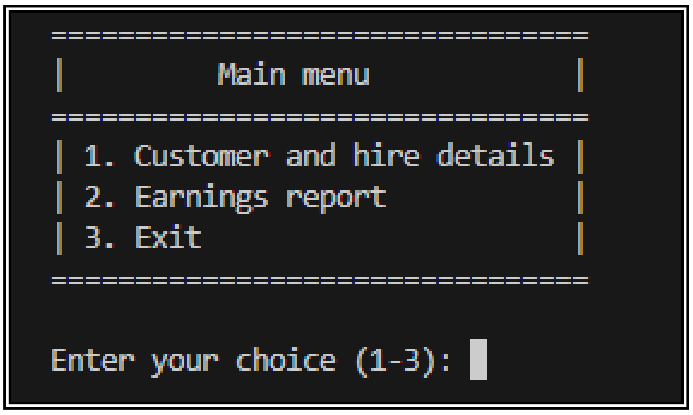
The program first checks whether the input is valid. If the input is invalid, the message "Invalid choice. Please enter 1, 2 or 3" is displayed and the user is redirected back to the menu. If the input is valid, the program displays a message based on the option chosen.

Option 1 displays "Customer & hire details selected" and returns to the menu.
Option 2 displays "Earnings report selected" and returns to the menu.
Option 3 displays "Exiting program" and ends the program.

Task 1 - Sample runs

The following screenshots shows that the program manages both valid and invalid inputs correctly.

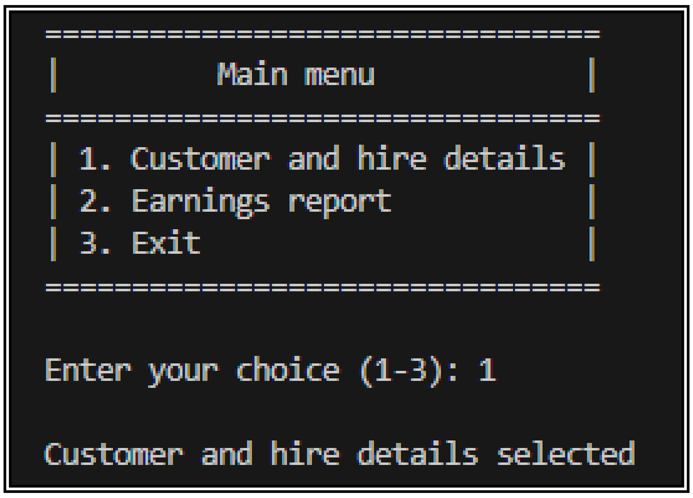
The return to main menu has not been shown in these examples, however additional screenshots in Appendix D, Figure 26 shows that the program returns to the menu after each option.



```
=================================
|          Main menu            |
=================================
| 1. Customer and hire details  |
| 2. Earnings report            |
| 3. Exit                       |
=================================

Enter your choice (1-3):
```

Figure 2 - Main menu

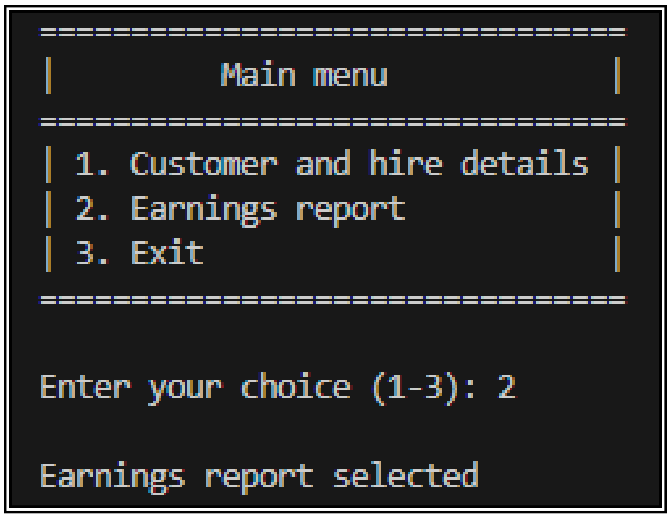
In Figure 2 the program starts by displaying the main menu. The user is prompted to enter a choice between 1 – 3.



```
=================================
|          Main menu            |
=================================
| 1. Customer and hire details  |
| 2. Earnings report            |
| 3. Exit                       |
=================================

Enter your choice (1-3): 1

Customer and hire details selected
```

Figure 3 - Option 1

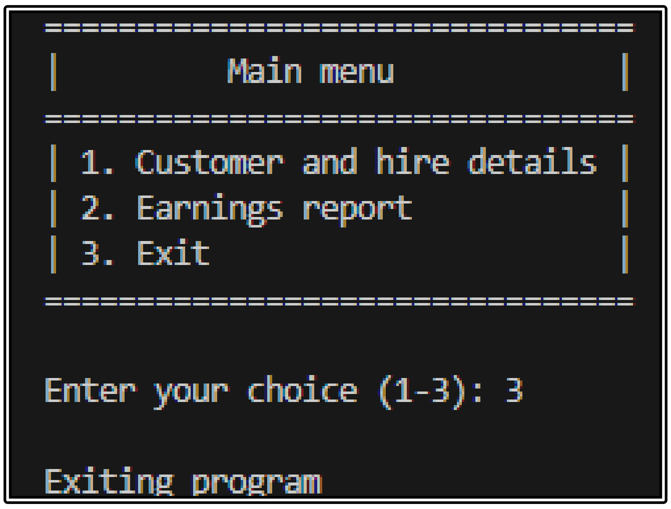
In Figure 3, the user selects option 1. The program outputs “Customer and hire details selected” confirming this option functions correctly.



```
==================================
|          Main menu             |
==================================
| 1. Customer and hire details |
| 2. Earnings report             |
| 3. Exit                        |
==================================


Enter your choice (1-3): 2

Earnings report selected
```

Figure 4 - Option 2

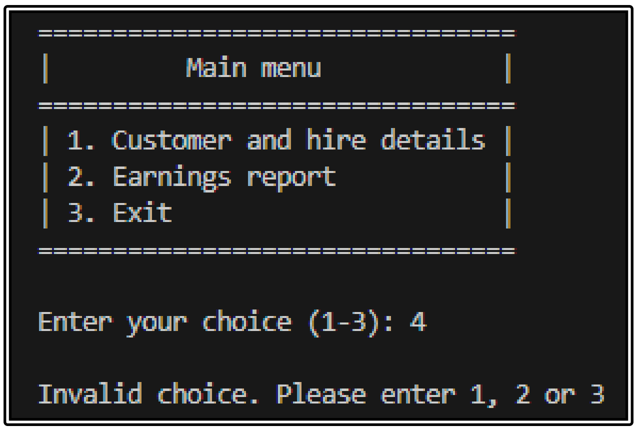
In Figure 4, the user selects option 2. The program outputs “Earnings report selected” confirming that this option functions correctly.



```
==================================
|          Main menu             |
==================================
| 1. Customer and hire details |
| 2. Earnings report             |
| 3. Exit                        |
==================================


Enter your choice (1-3): 3

Exiting program
```

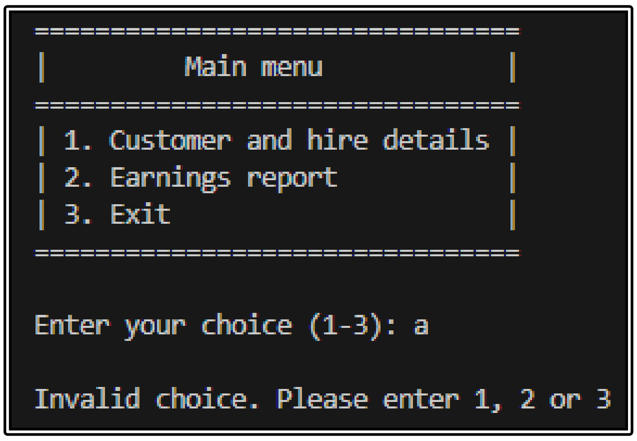
Figure 5 - Option 3

In Figure 5, the user selects option 3. The program outputs “Exiting program” and ends the program, confirming that this option functions correctly.



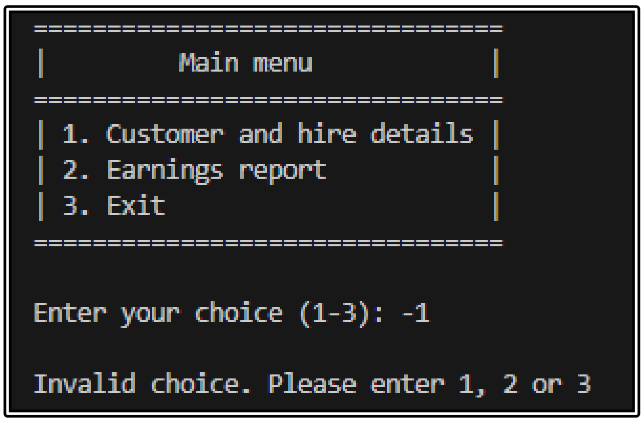
```
=================================
|         Main menu             |
=================================
| 1. Customer and hire details |
| 2. Earnings report           |
| 3. Exit                      |
=================================

Enter your choice (1-3): 4

Invalid choice. Please enter 1, 2 or 3
```

Figure 6 - Invalid input: 4



```
=================================
|         Main menu             |
=================================
| 1. Customer and hire details |
| 2. Earnings report           |
| 3. Exit                      |
=================================

Enter your choice (1-3): a

Invalid choice. Please enter 1, 2 or 3
```

Figure 7 - Invalid input: a



```
=================================
|         Main menu             |
=================================
| 1. Customer and hire details |
| 2. Earnings report           |
| 3. Exit                      |
=================================

Enter your choice (1-3): -1

Invalid choice. Please enter 1, 2 or 3
```

Figure 8 - Invalid input: -1

The figures above demonstrate how the program manages invalid inputs.

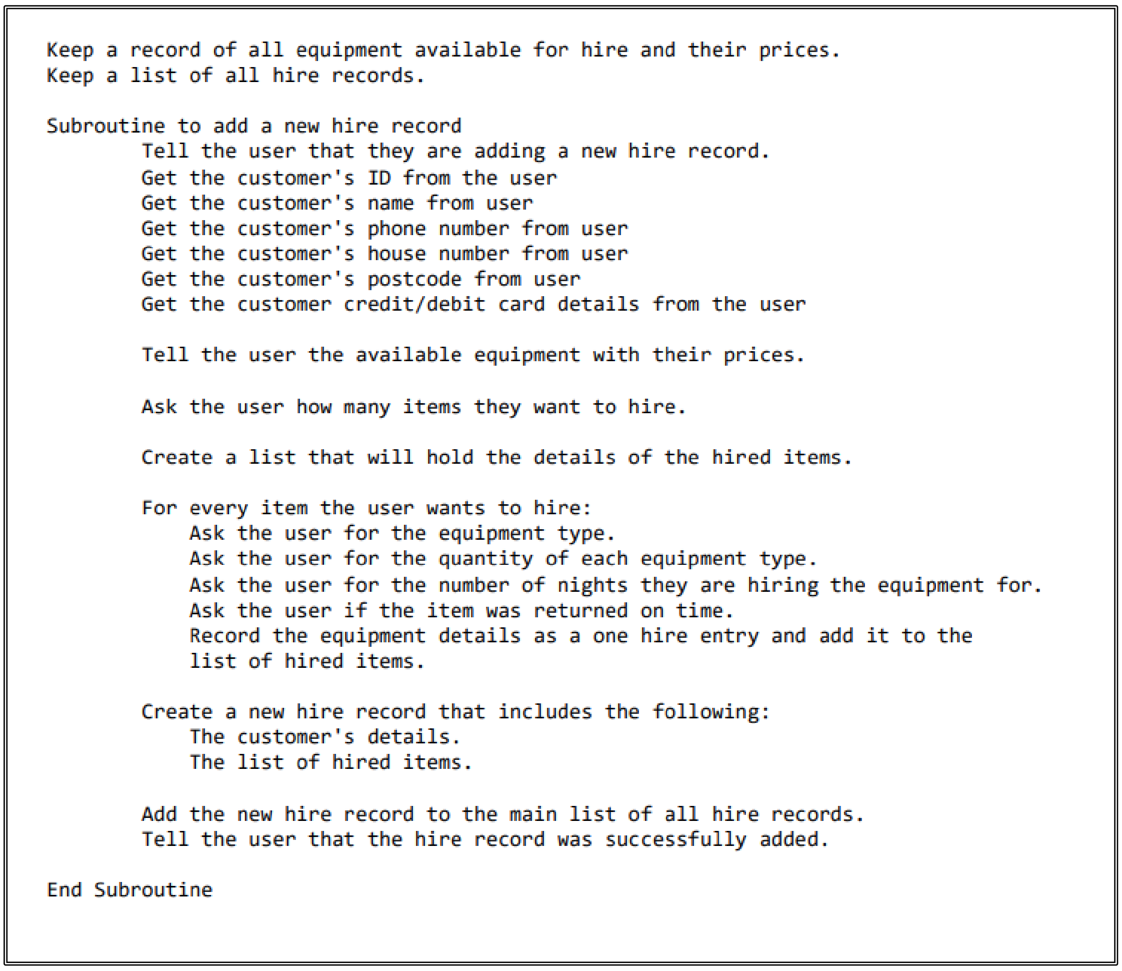
- In Figure 6, the user inputs “4”.
- In Figure 7, the user inputs “a”
- In Figure 8, the user inputs “-1”

In each case the program rejects the inputs and displays “Invalid choice. Please enter 1, 2 or 3” confirming that only options 1, 2 and 3 are accepted as valid inputs.

Task 2

Pseudocode

The full pseudocode text can be found in Appendix A



```
Keep a record of all equipment available for hire and their prices.
Keep a list of all hire records.

Subroutine to add a new hire record
        Tell the user that they are adding a new hire record.
        Get the customer's ID from the user
        Get the customer's name from user
        Get the customer's phone number from user
        Get the customer's house number from user
        Get the customer's postcode from user
        Get the customer credit/debit card details from the user

        Tell the user the available equipment with their prices.

        Ask the user how many items they want to hire.

        Create a list that will hold the details of the hired items.

        For every item the user wants to hire:
            Ask the user for the equipment type.
            Ask the user for the quantity of each equipment type.
            Ask the user for the number of nights they are hiring the equipment for.
            Ask the user if the item was returned on time.
            Record the equipment details as a one hire entry and add it to the
            list of hired items.

        Create a new hire record that includes the following:
            The customer's details.
            The list of hired items.

        Add the new hire record to the main list of all hire records.
        Tell the user that the hire record was successfully added.

End Subroutine
```

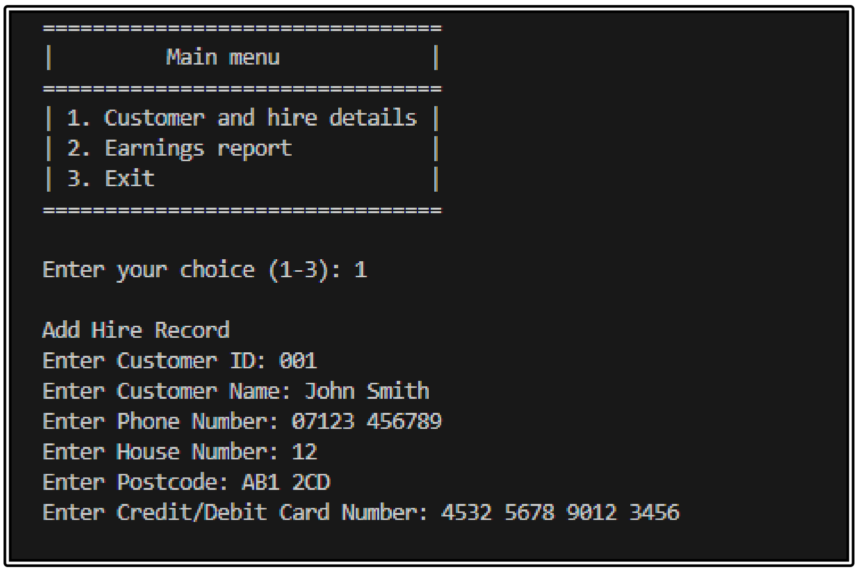
Figure 9 - Pseudocode

The pseudocode in Figure 9 starts by recording the available equipment and prices along with an empty list to store hire records. A subroutine is then created for adding a new hire record.

The subroutine starts by collecting customer details (ID, name, phone number, house number, postcode and payment details). The available equipment and their prices are displayed, and the user is asked how many items they would like to hire. For every item the user wants to hire, the program collects the equipment type, quantity, number of nights and if it was returned on time.

These details are stored as a hire entry and added to the list of hired items. A new hire record is created that contains both the customer’s details and the list of hired items. This record is added to the main list of all hire records and the user is displayed a message that the record was successfully added.

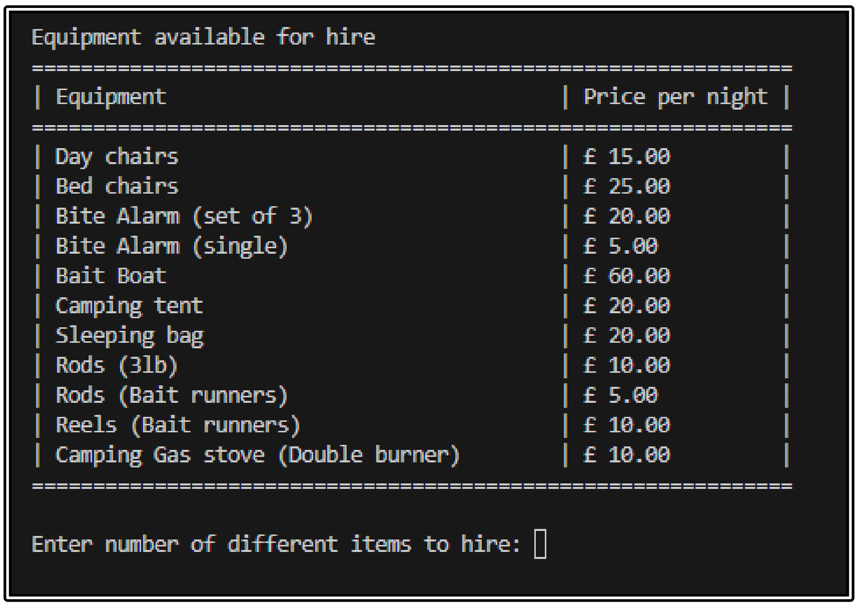
Task 2 of sample runs



```
================================
|         Main menu            |
================================
| 1. Customer and hire details |
| 2. Earnings report           |
| 3. Exit                      |
================================

Enter your choice (1-3): 1

Add Hire Record
Enter Customer ID: 001
Enter Customer Name: John Smith
Enter Phone Number: 07123 456789
Enter House Number: 12
Enter Postcode: AB1 2CD
Enter Credit/Debit Card Number: 4532 5678 9012 3456
```

Figure 10 - Customer detail input

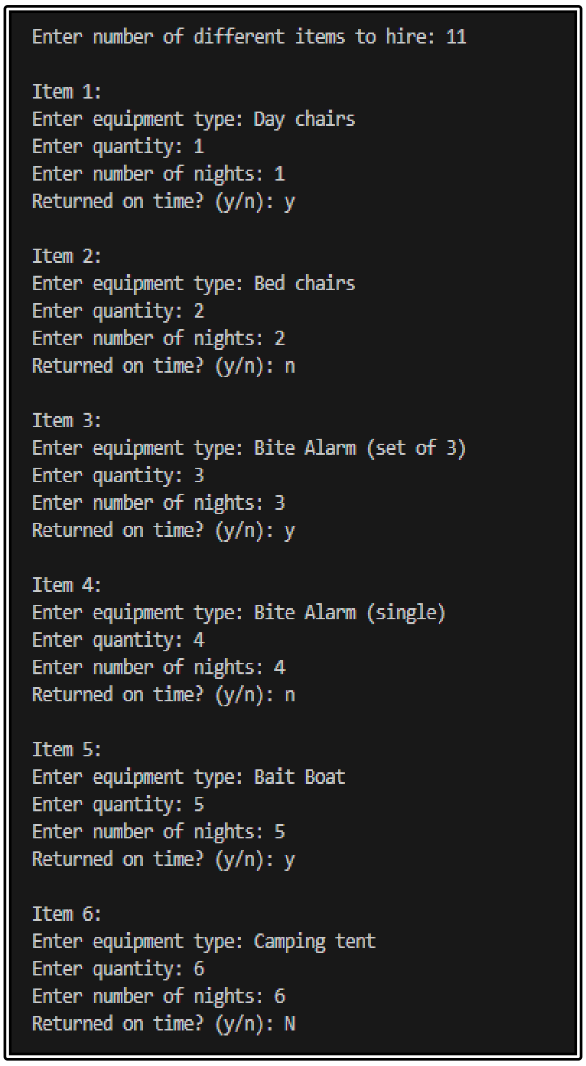
In Figure 10, the user selects option 1 from the menu. Instead of displaying the original message from Task 1, the program now calls the add hire record function. The program prompts the user for the customer’s details, including ID, name, phone number, house number, postcode and debit/credit card number. All the details are entered successfully, confirming the program correctly collects customer details before proceeding to next step.



```
Equipment available for hire
=============================================================
| Equipment                                | Price per night |
=============================================================
| Day chairs                               | £ 15.00         |
| Bed chairs                               | £ 25.00         |
| Bite Alarm (set of 3)                    | £ 20.00         |
| Bite Alarm (single)                      | £ 5.00          |
| Bait Boat                                | £ 60.00         |
| Camping tent                             | £ 20.00         |
| Sleeping bag                             | £ 20.00         |
| Rods (3lb)                               | £ 10.00         |
| Rods (Bait runners)                      | £ 5.00          |
| Reels (Bait runners)                     | £ 10.00         |
| Camping Gas stove (Double burner)        | £ 10.00         |
=============================================================

Enter number of different items to hire:
```

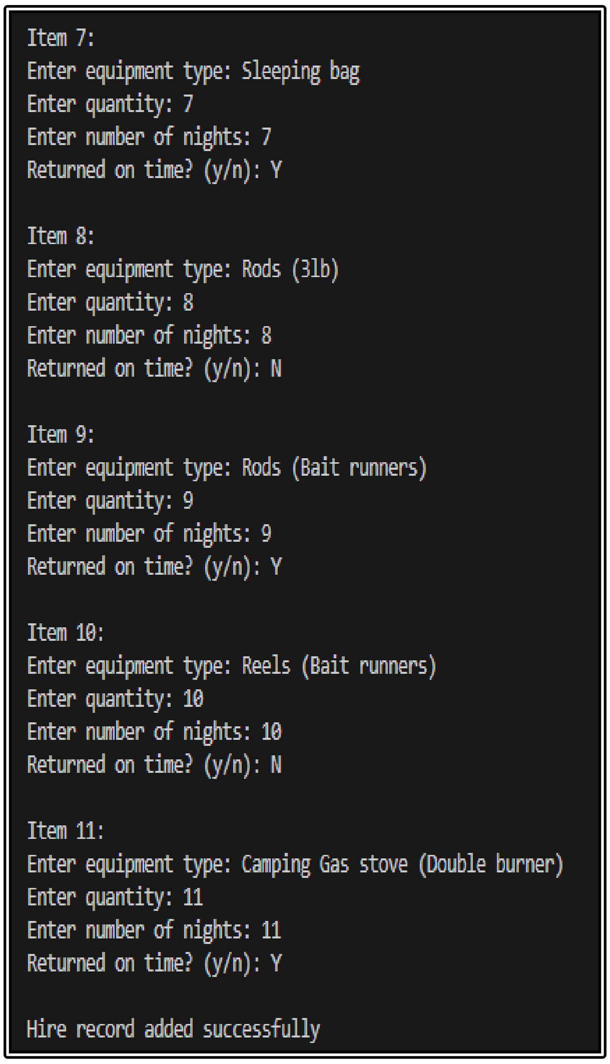
Figure 11 - Output available equipment

In Figure 11, once the customer’s details have been entered, the program displays a table of all available equipment with their prices. The user is then prompted to enter the number of different items they want to hire. This ensures the user has all the necessary information to make a decision.



```
Enter number of different items to hire: 11

Item 1:
Enter equipment type: Day chairs
Enter quantity: 1
Enter number of nights: 1
Returned on time? (y/n): y

Item 2:
Enter equipment type: Bed chairs
Enter quantity: 2
Enter number of nights: 2
Returned on time? (y/n): n

Item 3:
Enter equipment type: Bite Alarm (set of 3)
Enter quantity: 3
Enter number of nights: 3
Returned on time? (y/n): y

Item 4:
Enter equipment type: Bite Alarm (single)
Enter quantity: 4
Enter number of nights: 4
Returned on time? (y/n): n

Item 5:
Enter equipment type: Bait Boat
Enter quantity: 5
Enter number of nights: 5
Returned on time? (y/n): y

Item 6:
Enter equipment type: Camping tent
Enter quantity: 6
Enter number of nights: 6
Returned on time? (y/n): N
```

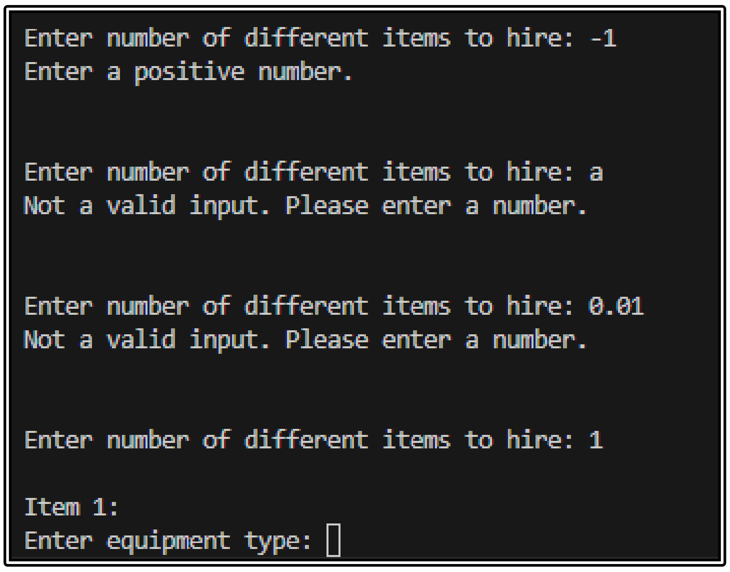
Figure 12 - Valid input for items 1-6



```
Item 7:
Enter equipment type: Sleeping bag
Enter quantity: 7
Enter number of nights: 7
Returned on time? (y/n): Y

Item 8:
Enter equipment type: Rods (3lb)
Enter quantity: 8
Enter number of nights: 8
Returned on time? (y/n): N

Item 9:
Enter equipment type: Rods (Bait runners)
Enter quantity: 9
Enter number of nights: 9
Returned on time? (y/n): Y

Item 10:
Enter equipment type: Reels (Bait runners)
Enter quantity: 10
Enter number of nights: 10
Returned on time? (y/n): N

Item 11:
Enter equipment type: Camping Gas stove (Double burner)
Enter quantity: 11
Enter number of nights: 11
Returned on time? (y/n): Y

Hire record added successfully
```

Figure 13 - Valid input for item 7-11

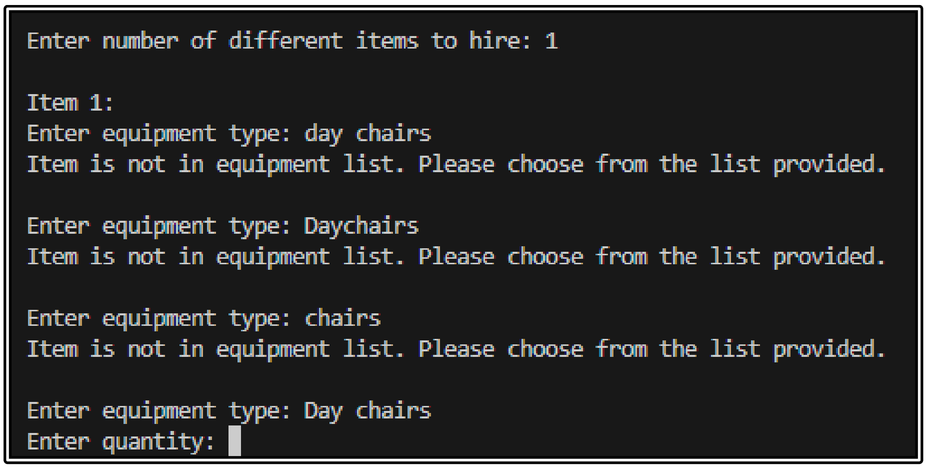
In Figure 12 and 13, the user enters the details for 11 different items, covering every type of equipment available. The quantity and number of nights increases by one for each item while the return status switches between y, n, Y, N to show input flexibility. Once all the details have been entered, the program displays “Hire record added successfully”, confirming that the hire record was successfully created and stored.



```
Enter number of different items to hire: -1
Enter a positive number.


Enter number of different items to hire: a
Not a valid input. Please enter a number.


Enter number of different items to hire: 0.01
Not a valid input. Please enter a number.


Enter number of different items to hire: 1

Item 1:
Enter equipment type: 
```

Figure 14 - Invalid inputs: Items for hire

In Figure 14, the user enters invalid values for the number of different items to hire, including -1, a and 0.01. Each invalid attempt results in an error message. When the input is not a number, the program displays "Not a valid input. Please enter a number.". For numerical but negative values, the program displays "Enter a positive number.". Only after a valid positive number is entered does the program proceed to the next step, confirming that the input validation works as intended.

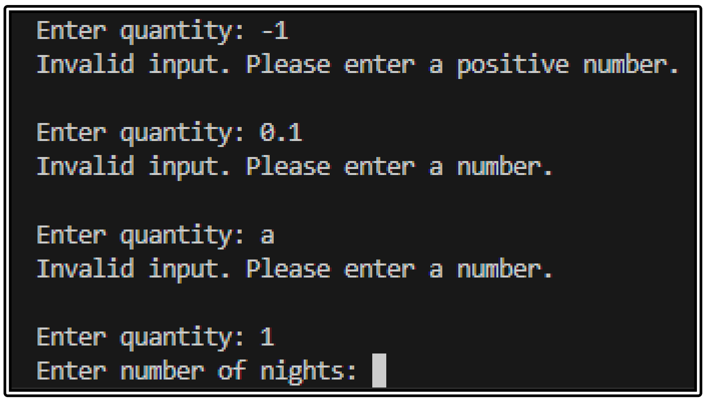


```
Enter number of different items to hire: 1

Item 1:
Enter equipment type: day chairs
Item is not in equipment list. Please choose from the list provided.

Enter equipment type: Daychairs
Item is not in equipment list. Please choose from the list provided.

Enter equipment type: chairs
Item is not in equipment list. Please choose from the list provided.

Enter equipment type: Day chairs
Enter quantity: 
```

Figure 15- Invalid inputs: Equipment type

In Figure 15, the user enters incorrect equipment names such as day chairs, Daychairs and chairs. Each invalid attempt results in the error message "Item is not in equipment list. Please choose from the list provided.". Only "Day chairs" is accepted confirming that the program validates equipment names accurately.

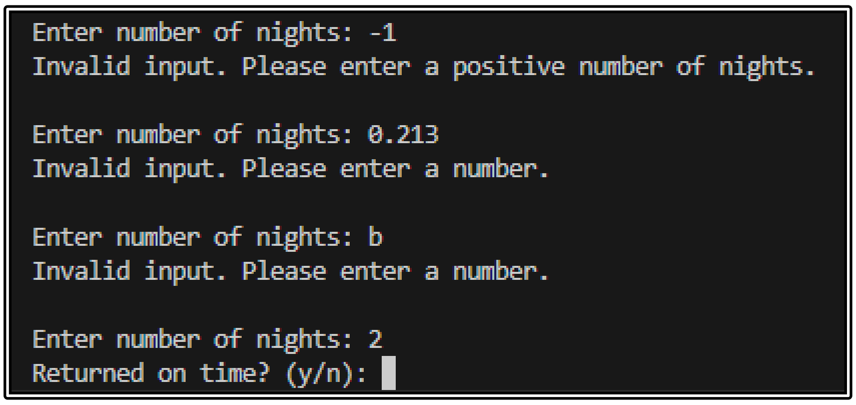
A potential improvement would be to make the equipment input more user friendly. Currently the program only accepts exact, case sensitive inputs, which increase the chance of typing errors. Providing each equipment with a related number would reduce input mistakes and make it more efficient.



```
Enter quantity: -1
Invalid input. Please enter a positive number.

Enter quantity: 0.1
Invalid input. Please enter a number.

Enter quantity: a
Invalid input. Please enter a number.

Enter quantity: 1
Enter number of nights: 
```

Figure 16- Invalid inputs: Quantity

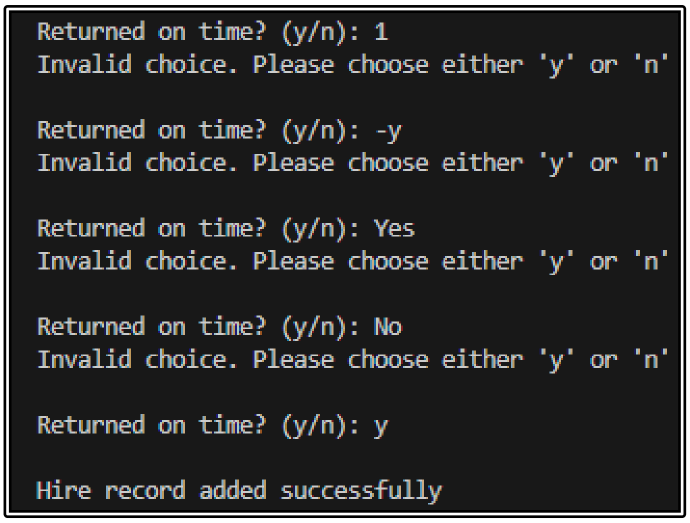
In Figure 16, the user enters invalid quantities such as -1, 0.1 and a. Each invalid attempt results in an error message. When the input is not a number, the program displays “Not a valid input. Please enter a number.”. For numerical but negative values, the program displays “Enter a positive number.”. Only after a valid positive number is entered does the program proceed to the next step, confirming that the input validation works as intended.



```
Enter number of nights: -1
Invalid input. Please enter a positive number of nights.

Enter number of nights: 0.213
Invalid input. Please enter a number.

Enter number of nights: b
Invalid input. Please enter a number.

Enter number of nights: 2
Returned on time? (y/n): 
```

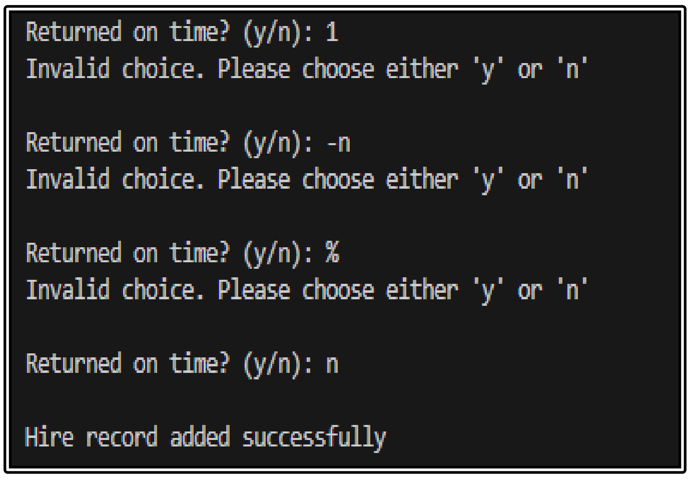
Figure 17- Invalid inputs: Number of nights

In Figure 17, the user enters invalid values for number of nights including -1, 0.213 and b. Each invalid attempt results in an error message, consistent with the input validation shown in Figures 14 and 16. Only when a positive number is entered, the program proceeds to next step.



```
Returned on time? (y/n): 1
Invalid choice. Please choose either 'y' or 'n'

Returned on time? (y/n): -y
Invalid choice. Please choose either 'y' or 'n'

Returned on time? (y/n): Yes
Invalid choice. Please choose either 'y' or 'n'

Returned on time? (y/n): No
Invalid choice. Please choose either 'y' or 'n'

Returned on time? (y/n): y

Hire record added successfully
```

Figure 18 - Invalid inputs: Returned on time (y)



```
Returned on time? (y/n): 1
Invalid choice. Please choose either 'y' or 'n'

Returned on time? (y/n): -n
Invalid choice. Please choose either 'y' or 'n'

Returned on time? (y/n): %
Invalid choice. Please choose either 'y' or 'n'

Returned on time? (y/n): n

Hire record added successfully
```

Figure 19 – Invalid inputs: Returned on time (n)

In Figures 18 and 19, the user enters invalid values for the returned on time input field, including 1, -y, -n, Yes, No and %. The program rejects these inputs and displays the error message “Invalid choice. Please choose either ‘y’ or ‘n’ ”Only after a valid input is provided, (y) or (n), does the program accept the input and complete the process by displaying “Hire record added successfully”

Task3

Flowchart

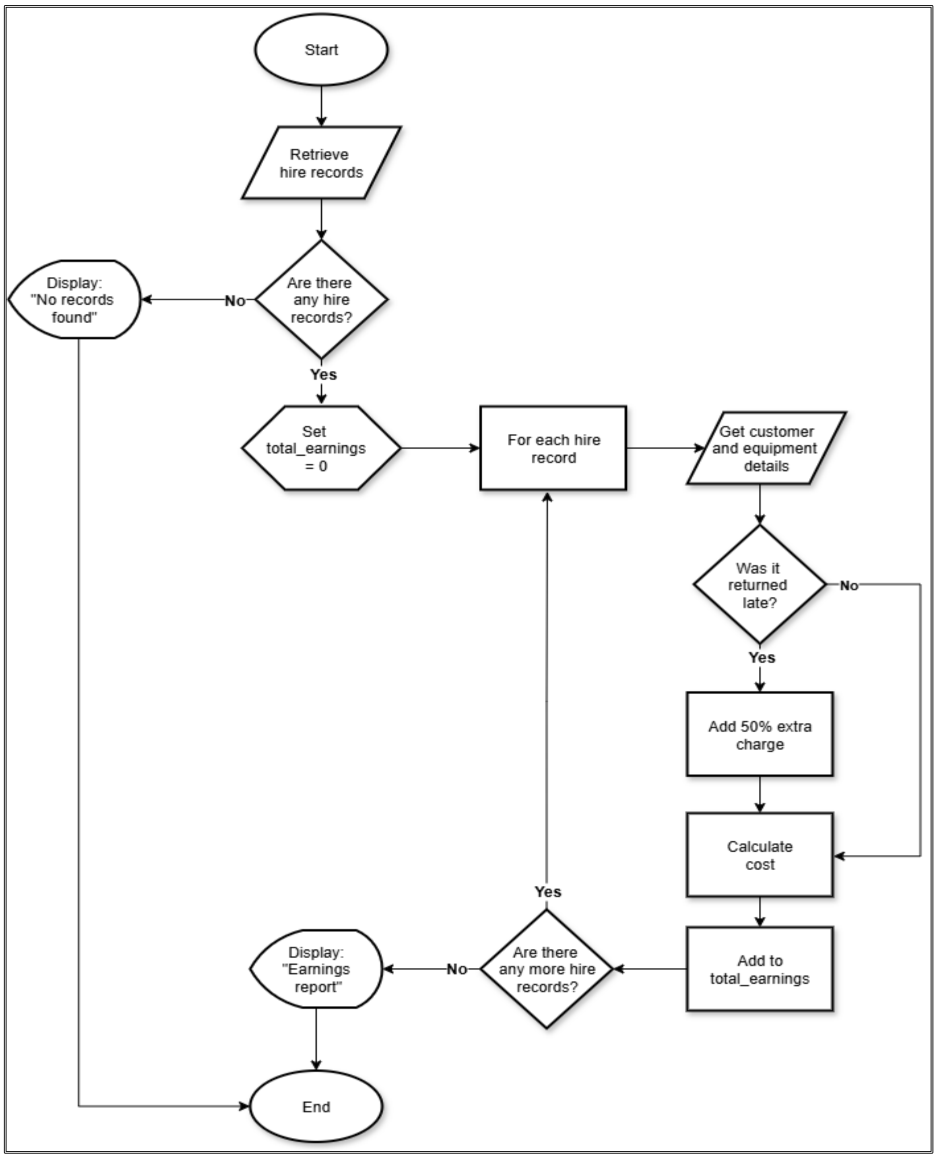


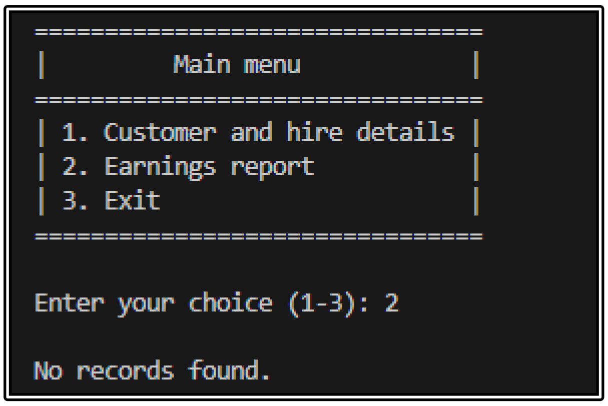
Figure 20 - Task 3 flowchart

The flowchart in Figure 20 represents the algorithm for generating the earnings report. The program starts by retrieving the hire records. If no record is found, the program displays the message “No records found” and ends.

If a record is found, the program sets the total earnings to zero and processes each hire record in turn. For every record, it retrieves the customer and equipment details then checks if the item was returned late. If items were returned late a 50% late fee is added before calculating the total cost. The cost is then added to the total earnings.

Then the program continues looping until all records have been processed. Once there are no more records to process, it displays the final earnings report and ends.

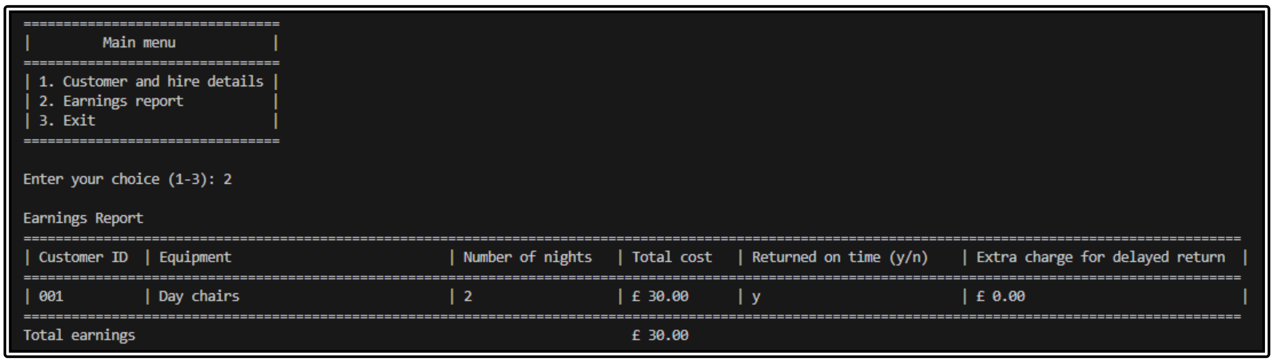
Task 3 sample runs



```
================================
|         Main menu            |
================================
| 1. Customer and hire details |
| 2. Earnings report           |
| 3. Exit                      |
================================

Enter your choice (1-3): 2

No records found.
```

Figure 21 - No records found

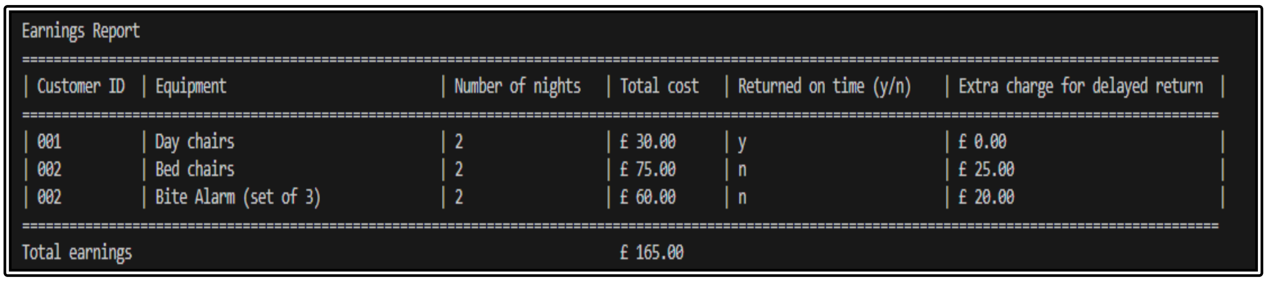
In Figure 21, the user selects option 2 from the menu to view the earnings report. Since no hire records have been created at this point, the program displays the message “ No records found”. This confirms that the program correctly handles the case where no data has been entered, instead of displaying an empty report, the user is provided with a clear message.



```
================================
|         Main menu            |
================================
| 1. Customer and hire details |
| 2. Earnings report           |
| 3. Exit                      |
================================

Enter your choice (1-3): 2

Earnings Report
=========================================================================================================================================================
| Customer ID  | Equipment                           | Number of nights   | Total cost   | Returned on time (y/n)    | Extra charge for delayed return  |
=========================================================================================================================================================
| 001          | Day chairs                          | 2                  | £ 30.00      | y                         | £ 0.00                           |
=========================================================================================================================================================
Total earnings                                                              £ 30.00
```

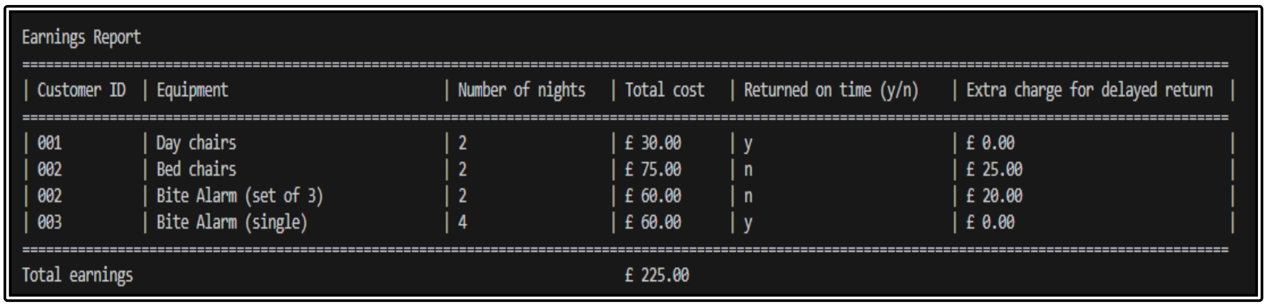
Figure 22 – Earnings report (1 Hire record)

In Figure 22, the user selects option 2 from the menu after one hire record has been created. The program retrieves the record and displays the earnings report, including the customer’s ID, equipment type, number of nights, total cost, if it was returned on time and extra charge for delayed returns. This confirms that the program generates, retrieves and displays a report when a single hire record is available.



```
Earnings Report
=====================================================================================================================
| Customer ID  | Equipment                    | Number of nights  | Total cost  | Returned on time (y/n)  | Extra charge for delayed return  |
=====================================================================================================================
| 001          | Day chairs                   | 2                 | £ 30.00     | y                       | £ 0.00                           |
| 002          | Bed chairs                   | 2                 | £ 75.00     | n                       | £ 25.00                          |
| 002          | Bite Alarm (set of 3)        | 2                 | £ 60.00     | n                       | £ 20.00                          |
=====================================================================================================================
Total earnings                                                  £ 165.00
```

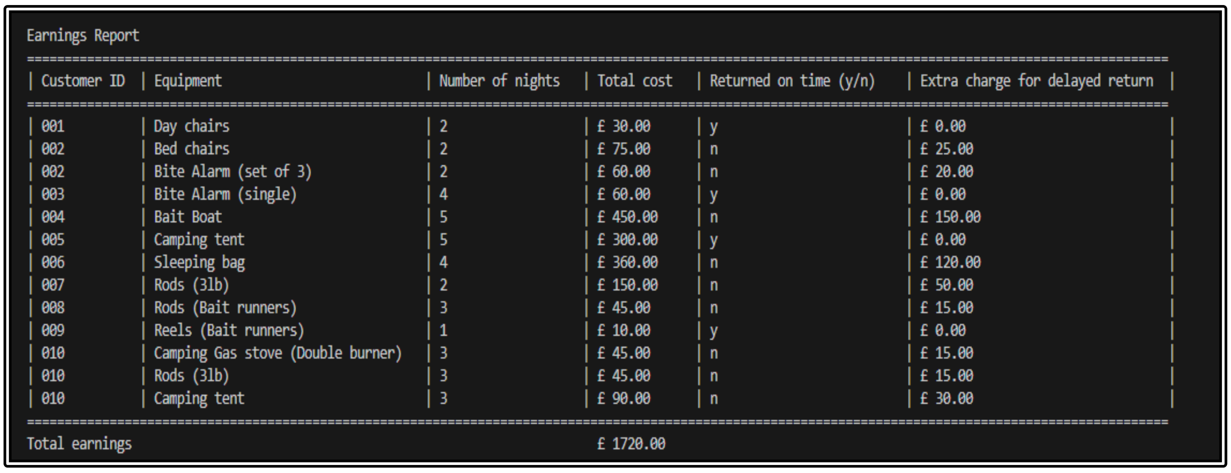
Figure 23 - Earnings report (2 Hire records)



```
Earnings Report
=====================================================================================================================
| Customer ID  | Equipment                    | Number of nights  | Total cost  | Returned on time (y/n)  | Extra charge for delayed return  |
=====================================================================================================================
| 001          | Day chairs                   | 2                 | £ 30.00     | y                       | £ 0.00                           |
| 002          | Bed chairs                   | 2                 | £ 75.00     | n                       | £ 25.00                          |
| 002          | Bite Alarm (set of 3)        | 2                 | £ 60.00     | n                       | £ 20.00                          |
| 003          | Bite Alarm (single)          | 4                 | £ 60.00     | y                       | £ 0.00                           |
=====================================================================================================================
Total earnings                                                  £ 225.00
```

Figure 24 – Earnings report (3 Hire records)

Figure 23 and 24 shows the earnings report after multiple hire records have been created. The program retrieves and displays these entries, including different equipment types hired by the same customer. This shows that the program can accurately calculate late fees for individual items, as customers can hire multiple items and return some on time while other are returned late. This confirms that the program can consistently retrieves and displays multiple records.



```
Earnings Report
=====================================================================================================================
| Customer ID  | Equipment                      | Number of nights  | Total cost  | Returned on time (y/n)  | Extra charge for delayed return  |
=====================================================================================================================
| 001          | Day chairs                     | 2                 | £ 30.00     | y                       | £ 0.00                           |
| 002          | Bed chairs                     | 2                 | £ 75.00     | n                       | £ 25.00                          |
| 002          | Bite Alarm (set of 3)          | 2                 | £ 60.00     | n                       | £ 20.00                          |
| 003          | Bite Alarm (single)            | 4                 | £ 60.00     | y                       | £ 0.00                           |
| 004          | Bait Boat                      | 5                 | £ 450.00    | n                       | £ 150.00                         |
| 005          | Camping tent                   | 5                 | £ 300.00    | y                       | £ 0.00                           |
| 006          | Sleeping bag                   | 4                 | £ 360.00    | n                       | £ 120.00                         |
| 007          | Rods (3lb)                     | 2                 | £ 150.00    | n                       | £ 50.00                          |
| 008          | Rods (Bait runners)            | 3                 | £ 45.00     | n                       | £ 15.00                          |
| 009          | Reels (Bait runners)           | 1                 | £ 10.00     | y                       | £ 0.00                           |
| 010          | Camping Gas stove (Double burner) | 3              | £ 45.00     | n                       | £ 15.00                          |
| 010          | Rods (3lb)                     | 3                 | £ 45.00     | n                       | £ 15.00                          |
| 010          | Camping tent                   | 3                 | £ 90.00     | n                       | £ 30.00                          |
=====================================================================================================================
Total earnings                                                  £ 1720.00
```

Figure 25 – Earnings report (10 Hire records)

In Figure 25, the earnings report displays ten hire records, including multiple customers, each with one or more pieces of equipment hired for multiple nights. Some items were returned late, while others were returned on time, this shows the program's ability to calculate late fees accurately.

This confirms that the program can handle large sets of data while keeping the report format consistent. It allows multiple entries to be stored and displayed without losing previous records and only clears the data when option 3 is selected from the main menu to end the program.

References

Balti, H. and Weiss, K. A., (2021). *Job Ready Python* [online]. Wiley.

Gaddis, T., (2024). *Starting Out with Python, Global Edition* [online]. Harlow: Pearson.

Horstmann, C. S., & Necaise, R. D., (2019). *Python For Everyone* [online]. Wiley.

Ståle, H. and Egil , J., (1998). *Python Tutorial*. Available from W3Schools: https://www.w3schools.com/python/ [Accessed 2025].

Yu, A., (n.d.). *100 Days of Code: The Complete Python Pro Bootcamp*. Available from Udemy: https://www.udemy.com/course/100-days-of-code/ [Accessed 2025].

Appendix A - Pseudocode

```
Keep a record of all equipment available for hire and their prices.
Keep a list of all hire records.

Subroutine to add a new hire record
    Tell the user that they are adding a new hire record.
    Get the customer's ID from the user
    Get the customer's name from user
    Get the customer's phone number from user
    Get the customer's house number from user
    Get the customer's postcode from user
    Get the customer credit/debit card details from the user

    Tell the user the available equipment with their prices.

    Ask the user how many items they want to hire.

    Create a list that will hold the details of the hired items.

    For every item the user wants to hire:
        Ask the user for the equipment type.
        Ask the user for the quantity of each equipment type.
        Ask the user for the number of nights they are hiring the equipment for.
        Ask the user if the item was returned on time.
        Record the equipment details as a one hire entry and add it to the list
        of hired items.

    Create a new hire record that includes the following:
        The customer's details.
        The list of hired items.

    Add the new hire record to the main list of all hire records.
    Tell the user that the hire record was successfully added.

End Subroutine
```

Appendix B – Task 1 Python Code

```
# Task 1: Main menu
# Student number: 24120579

# Program displays a menu and processes user's input.
# Loops until option 3 (Exit) is selected.
while True:
    # Displays main menu.
    print("=================================")
    print("|        Main menu           |")
    print("=================================")
    print("| 1. Customer and hire details |")
    print("| 2. Earnings report          |")
    print("| 3. Exit                     |")
    print("=================================")

    # Get user input.
    choice = input("\nEnter your choice (1-3): ")

    # Validates user input.
    if choice not in ["1", "2", "3"]:
        print("\nInvalid choice. Please enter 1, 2 or 3")
        continue

    # Process valid input.
    if choice == "1":
        print("\nCustomer and hire details selected")
    elif choice == "2":
        print("\nEarnings report selected")
    elif choice == "3":
        print("\nExiting program")
        break # Exits loop and ends the program.
```

Appendix C – Task 2 & 3 Python Code

```
# Task 2 and 3: Hiring equipment and Earnings report

# Student number: 24120579


# Dictionary containing available equipment and their prices.

equipment_inventory = {

    "Day chairs": {"price": 15.00},

    "Bed chairs": {"price": 25.00},

    "Bite Alarm (set of 3)": {"price": 20.00},

    "Bite Alarm (single)": {"price": 5.00},

    "Bait Boat": {"price": 60.00},

    "Camping tent": {"price": 20.00},

    "Sleeping bag": {"price": 20.00},

    "Rods (3lb)": {"price": 10.00},

    "Rods (Bait runners)": {"price": 5.00},

    "Reels (Bait runners)": {"price": 10.00},

    "Camping Gas stove (Double burner)": {"price": 10.00},

}


# List to store all hire records.

hires = []


# Subroutine to add a new hire record

def add_hire_record():

    print("\nAdd Hire Record")


    # Collects customer details.

    customer_id = input("Enter Customer ID: ")

    customer_name = input("Enter Customer Name: ")

    phone_number = input("Enter Phone Number: ")
```

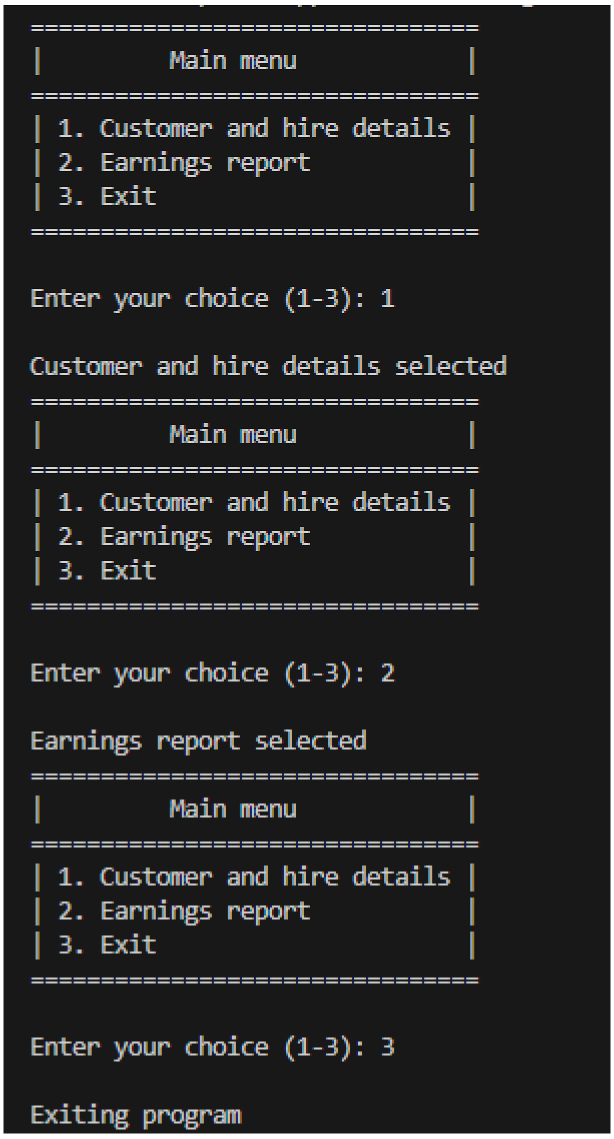
```
house_number = input("Enter House Number: ")
postcode = input("Enter Postcode: ")
credit_or_debit_card = input("Enter Credit/Debit Card Number: ")

# Displays available equipment and price.
print("\nEquipment available for hire")
# Creates table structure
print("="*62)
print(f"| {'Equipment':<40} | {'Price per night':<12} |")
print("="*62)

for equipment, details in equipment_inventory.items():
    print(f"| {equipment:<40} | £ {details['price']:<13.2f} |")

print("="*62)

# Collects and validates the number of items to hire.
while True:
    try:
        items_for_hire = int(input("\nEnter number of different items to hire: "))
        if items_for_hire > 0:
            break
        else:
            print("Enter a positive number.\n")
    except ValueError:
        print("Not a valid input. Please enter a number.\n")

# Collects hire item details.
hired_items = []

for item_number in range(1, items_for_hire + 1):
```

```
print(f"\nItem {item_number}: ")

# Collects and validates equipment type.
while True:
    equipment_type = input("Enter equipment type: ")
    if equipment_type in equipment_inventory:
        break
    else:
        print("Item is not in equipment list. Please choose from the list provided.\n")

# Collects and validates quantity.
while True:
    try:
        quantity = int(input("Enter quantity: "))
        if quantity > 0:
            break
        else:
            print("Invalid input. Please enter a positive number.\n")
    except ValueError:
        print("Invalid input. Please enter a number.\n")

# Collects and validates number of nights.
while True:
    try:
        number_of_nights = int(input("Enter number of nights: "))
        if number_of_nights > 0:
            break
        else:
            print("Invalid input. Please enter a positive number of nights.\n")
    except ValueError:
        print("Invalid input. Please enter a number.\n")
    # Collects and validates return status.
    while True:
        returned_on_time = input("Returned on time? (y/n): ").lower()
        if returned_on_time not in ["y", "n"]:
            print("Invalid choice. Please choose either 'y' or 'n'\n")
        else:
            break

    # Add item details to list.
    hired_items.append({
        "equipment_type": equipment_type,
        "quantity": quantity,
        "number_of_nights": number_of_nights,
        "returned_on_time": returned_on_time,
    })

# Create hire record from user entered details.
hire_record = {
    "customer_id": customer_id,
    "customer_name": customer_name,
    "phone_number": phone_number,
    "house_number": house_number,
    "postcode": postcode,
    "credit_or_debit_card": credit_or_debit_card,
    "hired_items": hired_items,
}

# Store hire record in hires list.
hires.append(hire_record)
print("\nHire record added successfully")
```

```
# Subroutine to generate the earnings report.
def earnings_report():
    # Checks if hire list is empty.
    if not hires:
        print("\nNo records found.")
        return

    total_earnings = 0

    # Creates earning report structure.
    print("\nEarnings Report")
    print("="*152)
    print(f"| {'Customer ID':<12} | {'Equipment':<35} | {'Number of nights':<18} | {'Total 
cost':<12} | {'Returned on time (y/n)':<25} | {'Extra charge for delayed return':<32} |")
    print("="*152)

    # Loops through every hire record.
    for record in hires:

        # Loops through every hired item in this record.
        for item in record["hired_items"]:
            equipment_type = item["equipment_type"]
            quantity = item["quantity"]
            number_of_nights = item["number_of_nights"]
            returned_on_time = item["returned_on_time"]

            # Calculates initial cost.
            price_per_night = equipment_inventory.get(equipment_type, {}).get("price", 0)
            cost = price_per_night * quantity * number_of_nights
            late_fee = 0
            # Calculate and apply late fee if items were returned late.
            if returned_on_time.lower() == "n":
                late_fee = cost*0.5
                cost += late_fee

            print(f"| {record['customer_id']:<12} | {equipment_type:<35} | {number_of_nights:<18}
| £{cost:< 11.2f} | {returned_on_time:<25} | £{late_fee:< 31.2f} |")

            # Add item cost to total earnings.
            total_earnings += cost

    print("=" * 152)
    # Displays final totals.
    print(f"{'Total earnings':<12} {'':<60} £{total_earnings: .2f}")

# Program displays a menu and processes user's input.
# Loops until option 3 (Exit) is selected.
while True:
    # Displays main menu.
    print("=================================")
    print("|         Main menu            |")
    print("=================================")
    print("| 1. Customer and hire details |")
    print("| 2. Earnings report           |")
    print("| 3. Exit                      |")
    print("=================================")

    # Get user input.
    choice = input("\nEnter your choice (1-3): ")
```

```
# Validates user input.
if choice not in ["1", "2", "3"]:
    print("\nInvalid choice. Please enter 1, 2 or 3")
    continue

# Process valid input.
if choice == "1":
    add_hire_record()
elif choice == "2":
    earnings_report()
elif choice == "3":
    print("\nExiting program")
    break # Exits loop and ends the program.
```

Appendix D - Addition Screenshots



```
================================
|         Main menu            |
================================
| 1. Customer and hire details |
| 2. Earnings report           |
| 3. Exit                      |
================================

Enter your choice (1-3): 1

Customer and hire details selected
================================
|         Main menu            |
================================
| 1. Customer and hire details |
| 2. Earnings report           |
| 3. Exit                      |
================================

Enter your choice (1-3): 2

Earnings report selected
================================
|         Main menu            |
================================
| 1. Customer and hire details |
| 2. Earnings report           |
| 3. Exit                      |
================================

Enter your choice (1-3): 3

Exiting program
```

Figure 26 - Return to main menu