Assignment (Jalan Technology Consulting)

Submitted By: - Harsh Sood

Question 1 –

```
#include <iostream>
using namespace std;

void rearrange(int arr[], int n)
{
   int max_idx = n - 1, min_idx = 0;
   int max_elem = arr[n - 1] + 1;
   for (int i = 0; i < n; i++)
   {
      if (i % 2 == 0)
      {
         arr[i] += (arr[max_idx] % max_elem) * max_elem;
         max_idx--;
      }
      else
      {
         arr[i] += (arr[min_idx] % max_elem) * max_elem;
```

```
            min_idx++;
        }
    }
    for (int i = 0; i < n; i++)
    {
        arr[i] = arr[i] / max_elem;
    }
}

int main()
{
    int arr[] = {2, 4, 6, 8, 10};
    int n = sizeof(arr) / sizeof(arr[0]);
    cout << "Original Arrayn";
    for (int i = 0; i < n; i++)
    {
        cout << arr[i] << " ";
    }
    rearrange(arr, n);
    cout << "\nModified Array\n";
    for (int i = 0; i < n; i++)
    {
        cout << arr[i] << " ";
    }
    return 0;
}
```

Question 2 –

```
class CoffeeMachine:

    running = False

    def __init__(self, water, milk, coffee_beans, cups, money):
        # quantities of items the coffee machine already had
        self.water = water
        self.milk = milk
        self.coffee_beans = coffee_beans
        self.cups = cups
        self.money = money

        #if the machine isnt running then start running
        if not CoffeeMachine.running:
            self.start()

    def start(self):
        self.running = True
        self.action = input("Write action (buy, fill, take, remaining, exit):\n")
        print()
        #possible choices to perform in the coffee machine
        if self.action == "buy":
            self.buy()
        elif self.action == "fill":
            self.fill()
        elif self.action == "take":
            self.take()
        elif self.action == "exit":
            exit()
        elif self.action == "remaining":
            self.status()

    def return_to_menu(self): # returns to the menu after an action
        print()
        self.start()

    def available_check(self): # checks if it can afford making that type of coffee at the moment
```

```
        self.not_available = "" # by checking whether the supplies goes below 0 after it is deducted
        if self.water - self.reduced[0] < 0:
            self.not_available = "water"
        elif self.milk - self.reduced[1] < 0:
            self.not_available = "milk"
        elif self.coffee_beans - self.reduced[2] < 0:
            self.not_available = "coffee beans"
        elif self.cups - self.reduced[3] < 0:
            self.not_available = "disposable cups"

        if self.not_available != "": # if something was detected to be below zero after deduction
            print(f"Sorry, not enough {self.not_available}!")
            return False
        else: # if everything is enough to make the coffee
            print("I have enough resources, making you a coffee!")
            return True

    def deduct_supplies(self): # performs operation from the reduced list, based on the coffee chosen
        self.water -= self.reduced[0]
        self.milk -= self.reduced[1]
        self.coffee_beans -= self.reduced[2]
        self.cups -= self.reduced[3]
        self.money += self.reduced[4]

    def buy(self):
        self.choice = input("What do you want to buy? 1 - espresso, 2 - latte, 3 - cappuccino, back - to main menu:\n")
        if self.choice == '1':
            self.reduced = [250, 0, 16, 1, 4] # water, milk, coffee beans, cups, money
            if self.available_check(): # checks if supplies are available
                self.deduct_supplies() # if it is, then it deducts

        elif self.choice == '2':
            self.reduced = [350, 75, 20, 1, 7]
            if self.available_check():
                self.deduct_supplies()

        elif self.choice == "3":
            self.reduced = [200, 100, 12, 1, 6]
            if self.available_check():
                self.deduct_supplies()
        elif self.choice == "back": # if the user changed his mind
            self.return_to_menu()

        self.return_to_menu()

    def fill(self): # for adding supplies to the machine
        self.water += int(input("Write how many ml of water do you want to add:\n"))
        self.milk += int(input("Write how many ml of milk do you want to add:\n"))
        self.coffee_beans += int(input("Write how many grams of coffee beans do you want to add:\n"))
        self.cups += int(input("Write how many disposable cups of coffee do you want to add:\n"))
        self.return_to_menu()

    def take(self): # for taking the money from the machine
        print(f"I gave you ${self.money}")
        self.money -= self.money
        self.return_to_menu()
    def status(self): # to display the quantities of supplies in the machine at the moment
        print(f"The coffee machine has:")
        print(f"{self.water} of water")
        print(f"{self.milk} of milk")
        print(f"{self.coffee_beans} of coffee beans")
        print(f"{self.cups} of disposable cups")
        print(f"${self.money} of money")
        self.return_to_menu()


CoffeeMachine(400, 540, 120, 9, 550) # specify the quantities of supplies at the beginning
            # water, milk, coffee beans, disposable cups, money
```