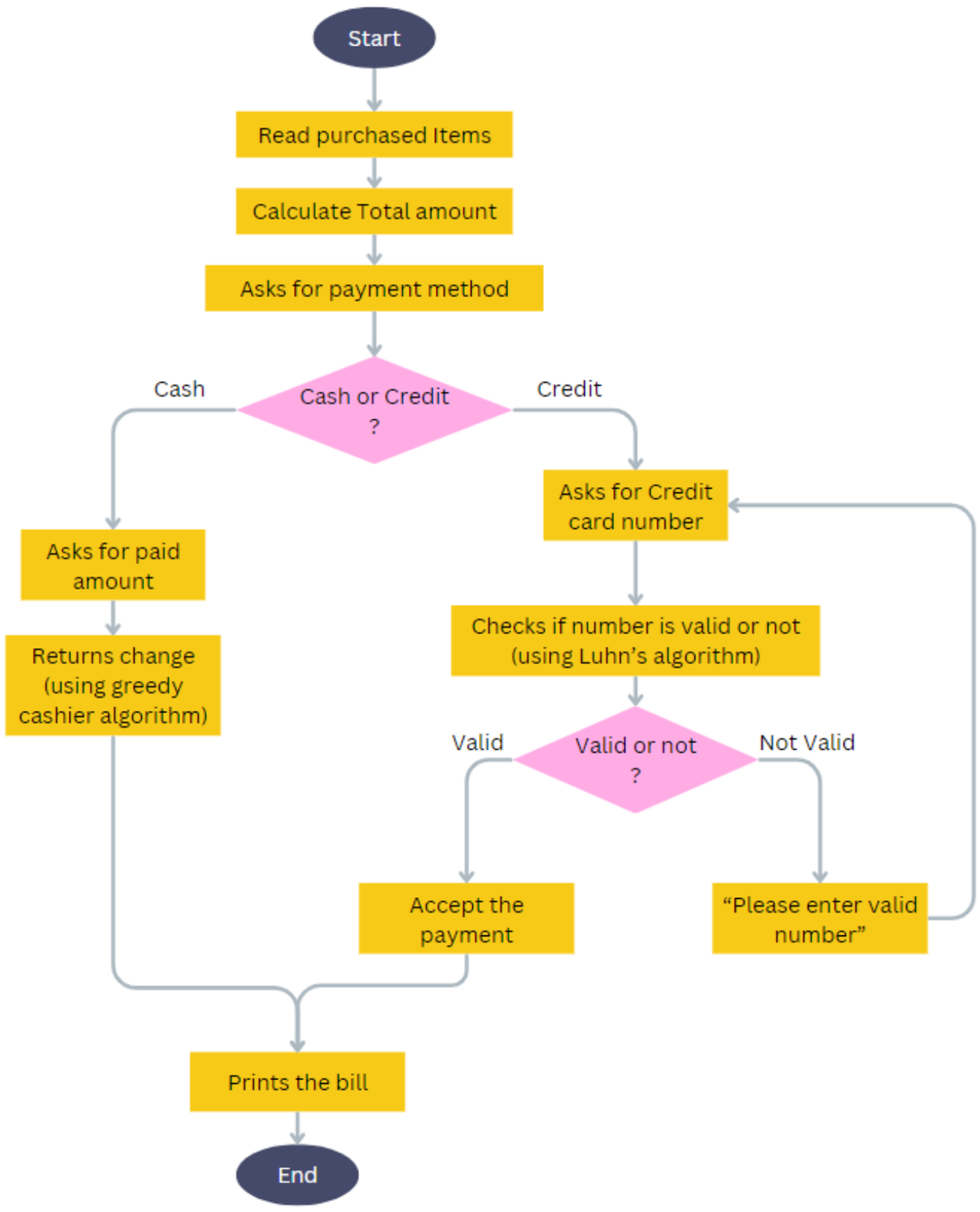
A. D. Patel institute of Technology
Computer Engineering
A.Y.: 2023-24, 1st semester
Computer Programming with C - 202000110

Project Definition: Making a cash counter that count cash, verify credit card and makes bill

Project Description: This is a project where you can enter purchased items and it'll calculate total amount and you can pay the bill with either cash or credit card. If payment done with cash it'll return remaining amount and will print the bill.



Screenshots of output:

```
item name: pen
item name: paper
item name: pencil
item name: pouch
item name: skybag
item name: end
Total bill: 842

How would you like to pay? (cash or credit): cash
Amout paid: 1000

Cash counter will return 5 (coins + notes)
100
50
5
2
1


_____________________________
Item:           Prise

pen:            10
paper:          2
pencil:         5
pouch:          75
skybag:         750

Total:          842
_____________________________
```

```
How would you like to pay? (cash or credit): credit

Enter credit card number: 378282246310006
INVALID
Please enter valid number.

Enter credit card number: 378282246310005
AMEX
Amount 842 has been paid.
```

Luhn's algorithm

1. Multiply every other digit by 2, starting with the number's second-to-last digit, and then add those products' digits together.
2. Add the sum to the sum of the digits that weren't multiplied by 2.
3. If the total's last digit is 0 (or, put more formally, if the total modulo 10 is congruent to 0), the number is valid!

That's kind of confusing, so let's try an example with David's Visa: 4003600000000014.

1. For the sake of discussion, let's first underline every other digit, starting with the number's second-to-last digit:

   4003600000000014

   Okay, let's multiply each of the underlined digits by 2:

   1•2 + 0•2 + 0•2 + 0•2 + 0•2 + 6•2 + 0•2 + 4•2

   That gives us:

   2 + 0 + 0 + 0 + 0 + 12 + 0 + 8

   Now let's add those products' digits (i.e., not the products themselves) together:

   2 + 0 + 0 + 0 + 0 + 1 + 2 + 0 + 8 = 13

2. Now let's add that sum (13) to the sum of the digits that weren't multiplied by 2 (starting from the end):

   13 + 4 + 0 + 0 + 0 + 0 + 0 + 3 + 0 = 20

3. Yup, the last digit in that sum (20) is a 0, so David's card is legit!

Greedy Cashier algorithm:

According to the National Institute of Standards and Technology (NIST), a greedy algorithm is one "that always takes the best immediate, or local, solution while finding an answer. Greedy algorithms find the overall, or globally, optimal solution for some optimization problems, but may find less-than-optimal solutions for some instances of other problems."

What's all that mean? Well, suppose that a cashier owes a customer some change and in that cashier's drawer 500Rs, 200Rs, 100Rs, 50Rs, 20Rs, 10Rs, 5Rs, 2Rs, 1Rs. The problem to be solved is to decide which coins and how many of each to hand to the customer. Think of a "greedy" cashier as one who wants to take the biggest bite out of this problem as possible with each note they take out of the drawer. For instance, if some customer is owed 73Rs, the biggest first bite that can be taken is 50. (That bite is "best" as much as it gets us closer to 0Rs faster than any other note would.) Note that a bite of this size would whittle what was a 73Rs problem down to a 23Rs problem, since 73 - 50 = 23. That is, the remainder is a similar but smaller problem. Needless to say, another 50Rs bite would be too big (assuming the cashier prefers not to lose money), and so our greedy cashier would move on to a bite of size 20Rs, leaving him or her with a 3Rs problem. At that point, greed calls for one 2Rs bite followed by one 1Rs bite, at which point the problem is solved. The customer receives one 50Rs, one 20Rs, one 2Rs, and one 1Rs: four notes/coins in total.

It turns out that this greedy approach (i.e., algorithm) is not only locally optimal but also globally so for America's currency (and also the European Union's). That is, so long as a cashier has enough of each coin, this largest-to-smallest approach will yield the fewest coins possible.

Conclusion: Cash counter is used to accurately and efficiently count and handle cash. It collects payment by accepting cash/credit card from customer and makes change for customer.

Reference:
https://cs50.harvard.edu/x/2023/
https://cs50.harvard.edu/x/2023/psets/1/cash/
https://cs50.harvard.edu/x/2023/psets/1/credit/

Name	Enrollment no.
Palak Sondarva	1230204060104 2
Rachana Solanki	1230204060104 7
Rudra Sharma	1230204060105 0