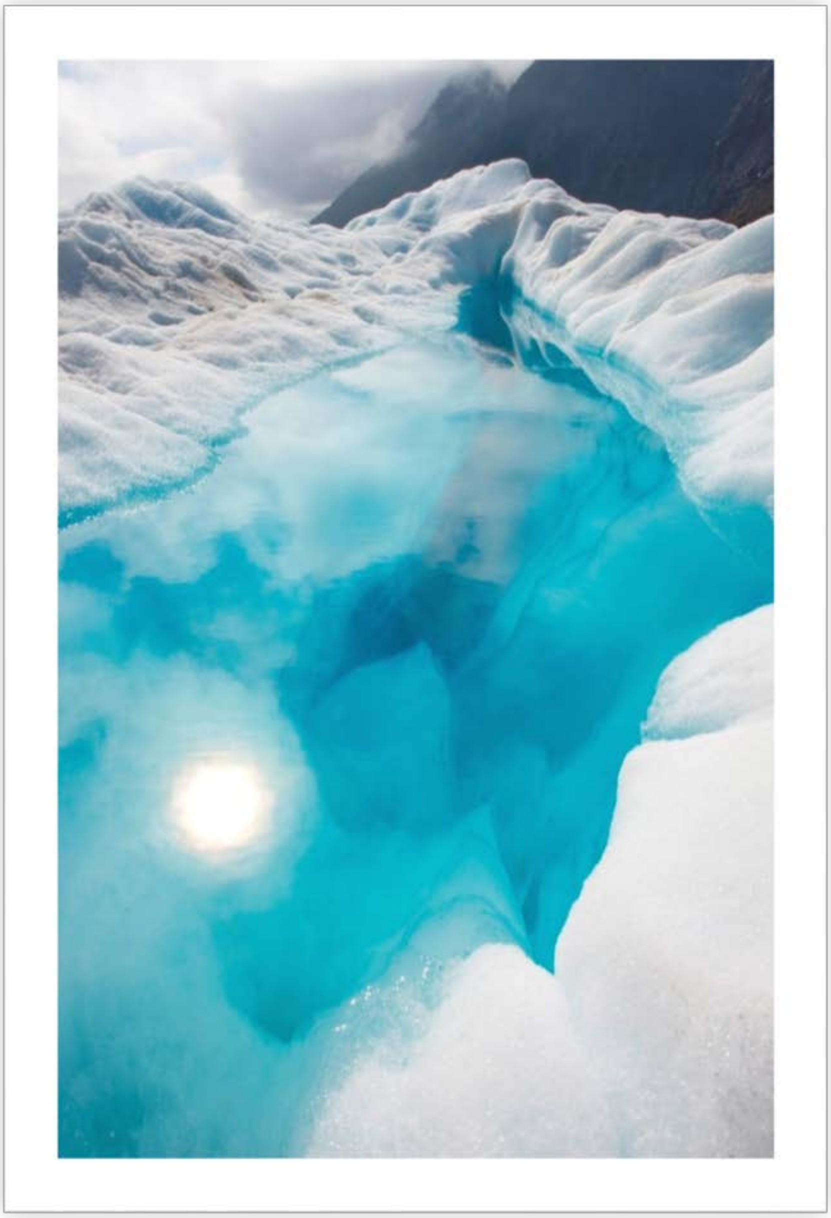


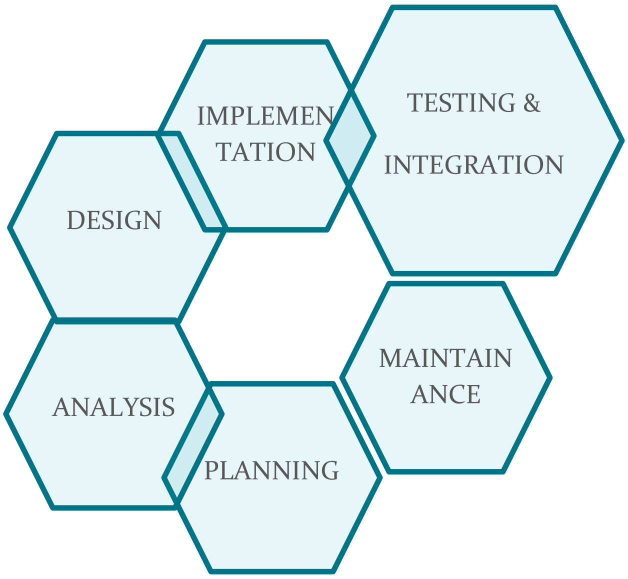
BUSINESS ANALYSIS

AND MAKING OF A RECORD KEEPING PROGRAM

JNV Khowai | Computer Science | Autumn 2024

*THIS DESIGN CHOICE OFFERS BETTER USR EXP

THE DEVELOPMENT CYCLE



i.

INTRODUCTION

As part of my autumn work, this project contains a study of my Brother's business. Certain aspects like the dynamics of the transactions has been ignored and is only concerned will all the (typically) busy calculations involved.

Use of AI was only partly done just for the research that had to be done to cope up with all the technicalities (& features) of python. (no code was generated)
The rest of the research was done only through the python documentation.

Passionately,
by Dhrubajyoti Chowdhury

INDEX

ii.

MY BROTHER’S BUSINESS

My brother, along with being passionate about blogging also does business. His business work can be classified in following way:

1. Taking orders
2. Keeping stocks of goods
3. Providing timely deliveries
4. Keeping record of all transactions

The first three of the categories involve dynamic aspects of his business: he has to take calls, tell his customer what is in the stock, Inform the customer all other information like change in price and demand in market of the concerned good.

Though the fourth category is comparatively less involved. But still having a software that can handle somewhat of the record keeping work shall make the process less time consuming and efficient; say in terms of how many buttons are pressed when compared with a use of normal calculator.

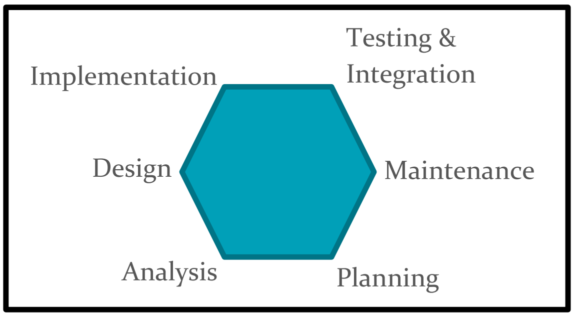
> Me: “What features do you think would be useful for all the calculations of your stuff ?”
>
> Bro: “Maybe it could calculate cost and profit…”

iii.

THE DEVELOPMENT CYCLE

The development cycle is a very handy rule of thumb which has simple sequence of steps (in fig) that help in the development a program.

The development cycle was iterated through many times before the current implementation. The following steps explained belong to the last iteration of the development cycle.



a.

PLANNING

Our program is going to handle records of transactions and provide useful methods of manipulating data.

b.

ANALYIS

After talking to him and looking through my brother’s Note Book, the things in notice were as follows:
The records generally had information relating to items and customers; there were different records – those for frequent transactions, and those for monthly.

So on a whole, our program needs to get and manipulate varied data.

c.

DESIGN

It’s simplest make a CLI.
Our program is going to offer the user a list of commands that the user can use for updating records and extracting useful information.

1. ‘make’ to update record
2. ‘sum’ to calculate values like total, median etc.
3. ‘rcrd’ to view what’s in the record
4. ‘sell’ to add additional information*

‘q’ to quit from current scope.

*THIS DESIGN CHOICE OFFERS BETTER USR EXP

*This is a design choice that facilates usr exp

d.

IMPLEMENTATION

After a few iterations of the development cycle, I was facing problems that needed clever implementation of my idea. So I read through (downloaded) python’s documentation that and figure out why I was getting certain kinds of errors and that in turn helped me discover certain features of python like:

the use of ‘yield’,

the fact that variables that are declared outside a scope are shadowed inside that scope

and more.

While trying to make my code more readable, I stumbled upon the concept of ‘Functional Programming’ and ‘Pure Functions’.
I adopted this into my code and also reconsidered the data structure which went from having like 7 variables to just a single dictionary.

The following section break-downs my code:

```
log = "Enter \n\
   1. 'make' to enter key value pairs \n\
   2. 'sum' to calculate statistical values \n\
   3. 'rcrd' to show records \n\
   4. 'sell' to enter selling details \n\
    *COST_PRICE stands for that of per piece \n\
    *SELLING_PRICE stands for that of per piece \n\
   'q' to quit from current scope \n\
       "
print(log)
```

- This lets the user know about the program.

```
rcrd = {} # everything
# update record
def make(rc = {}):

  while True:

     a = input(" ITEM_NAME: ")#press q when asked
    if a == 'q':
       print()
       break
```

...

...

```
     rcrd[a] = [0, 0, 0]

     rc[a][0] = float(input ("COST_PRICE: "))

     rc[a][1] = float(input(" QUANTITY: "))

     rc[a][2] = 0

     print()

  return rc # we can get all info from here
```

- This is the make() function which repeatedly takes three values from the usr viz. ITEM_NAME, COST_PRICE, and QUANTITY.(fourth, sp is set 0)
- It takes a dictionary as an argument and returns the same updated.
- The data structure is clear here, ITEM_NAME is the key and others its values as a list.

```
# calculate statistical values
def sum(rc = {}):

  total_value = 0
  total_qty = 0
  items = len(rc)
  value_list = []

  for key in rc:

     value_list.append(rc[key][0]*rc[key][1])
     total_qty += rc[key][1]
     total_value += rc[key][0]*rc[key][1]


  mean = total_value/total_qty

  # middle value in list
  def m(b=0):
     return (sorted(value_list))[int(items/2 +0.5) -1]
  if items %2 == 0:
     median = (m(-0.5) + m(0.5))/2
  else:
     median = m()

  return [total_value, mean, median]
```

- sum() Returns total, mean and median value.

```
def sell(rc = {}):

    print("Enter your selling costs: \n")
    a = 1

    for key in rc:
        print(str(a) +". "+ str.rjust(key, 8), end = ': ')

        rc[key][2] = float(input())

        a += 1
    print("\nDone!\n")

    return rc
```

- sell() takes selling costs as additional information.

```
#----------main:
while True:

    cmd = str.lower(input("Command: "))
    print()

    if cmd == 'make':
        rcrd = make(rcrd) # updating records

    elif cmd == 'sum':
        rtrn_vals = sum(rcrd)

        def a(b):
            return str.rjust("{:.2f}".format(rtrn_vals[b]), 10)
        print(
            " Total value:", a(0),
          "\n  mean value:", a(1),
          "\nmedian value:", a(2),'\n'
            )

    elif cmd == 'sell':
        rcrd = sell(rcrd)

    elif cmd == 'rcrd':

        a = 1
        for item in rcrd:        # print rcrd
            cp = rcrd[item][0]*rcrd[item][1]
            sp = rcrd[item][2]*rcrd[item][1]
            p = "{:.2f}".format(sp - cp)
            CP = "{:.2f}".format(cp)

            print(a,'.', str.rjust(item +"  "+ CP, 15) +
str.rjust("  P: "+ p, 15))
            a += 1
        print("\n\'P\' stands for profit. ")
        print()

    elif cmd == 'q':
        break
```

...

...

- Finally the whole thing is integrated Into a single piece here.

TESTING & INTEGRATION

With repetitive brainstorming, ideas were gathered to solve problems of string formatting and more generally the correctness of code.

MAINTENANCE

The whole project is meant to be FOSS based. To achieve this, this project has been uploaded to the “github” platform:

https://github.com/Dhrubajyoti930/School-projects-/

SAMPLE OUTPUT

```
Enter
     1. 'make' to enter key value pairs
     2. 'sum' to calculate statistical values
     3. 'rcrd' to show records
     4. 'sell' to enter selling details
      *COST_PRICE stands for that of per piece
      *SELLING_PRICE stands for that of per piece
     'q' to quit from current scope

Command: make

 ITEM_NAME: Pen
COST_PRICE: 3.5
 QUANTITY: 100

 ITEM_NAME: DOMS pencil
COST_PRICE: 15.5
 QUANTITY: 210

 ITEM_NAME: q

Command: sum

 Total value:    3605.00
  mean value:      11.63
median value:     350.00

Command: sell

Enter your selling costs:

1.      Pen: 5
2. DOMS pencil: 22

Done!

Command: rcrd

1 .     Pen  350.00       P: 150.00
2 . DOMS pencil  3255.00      P: 1365.00

'P' stands for profit.

Command: q
```