Zachary Meisner

IT 140

May 22, 2021

Professor Carter

What input does the computer need?

The computer needs -

- Access to the company bank in order to move the desired amount of money that is appropriate from A to B
- Access to the employee's bank account (for direct deposit) or perhaps some valid way to make and print checks and mail them to the employee
- How many hours did the employee work?
- What week this took place
- What is overtime?
- What is regular hours (0-40)
- How to distinguish the final variable the employee earned

What steps does the program need to follow to process the input? What output?

START

- Company_bank
    - Pass
- Employee_bank
    - Pass
- Current_week
    - pass
- Act_Paycheck = 0
-
- Test_paycheck = 0

-
- Test_hours = 0

QUESTION 1

- employee_hours = input('Enter Hours Worked')
-
- **PROCESS 1**

If employee_hours <= 40

- Test_hours = Employee_hours * 20

Elif employee_hours >= 60

employee_hours – 40

Employee_hours * 30

Test_hours = Employee_hours + 800

Else:

Print('Please Enter Proper Input')

QUESTION 2

input('So and so made', Test_hours, 'during the week of', current_week, 'Does this look correct?')

PROCESS 2

If yes

Test_paycheck = Test_hours

Elif no

return

Else

Print('please enter an input I understand')

As you can see above, we might need decision branching when making decisions of where to move numbers around and how to compute certain variables and distinguish between different types of input to get specified outputs for the system.

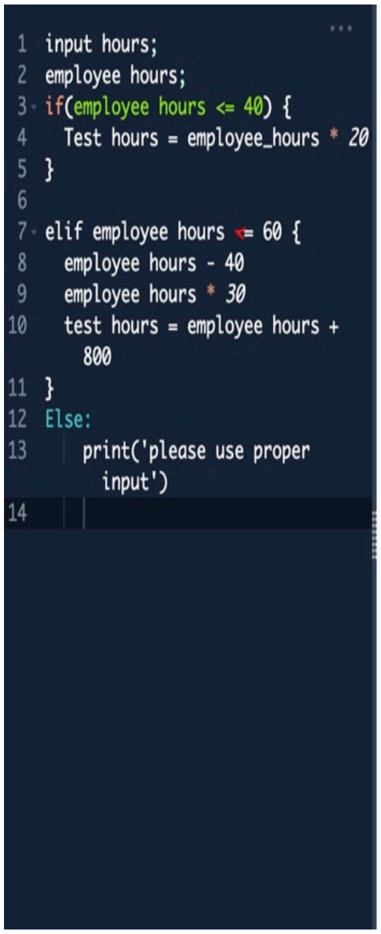
I think it would be nearly impossible to account for every possible value and to do so might be a bad way to program this system because we want something that is not easy to break.

What I mean by this is that I think that there is a lot of really great possibilities when it comes to object oriented programming, and really understanding the functions of specific classes and the way they interact and "talk" with one another is important. Alternatively, if we were to use the if elif else data structure for every instance than it would really suck for the user if they put in the wrong input, had to go back to a specified spot, have to distinguish which exact specified spot. Blah blah

To get to the point of what I was just pointing out, the UX design for the program would be absolutely terrible and nobody would want to use it because people, although it is great to have that many options, may find it difficult to train people on these types of systems in the case they have so many options or complications.

Maybe for someone that is good at this type of thing, it would be no issue, but if a bank for example was trying to hire someone that was not good at computers due to their lack of available qualified staff due to the location the bank is set in, this might not be the greatest idea.

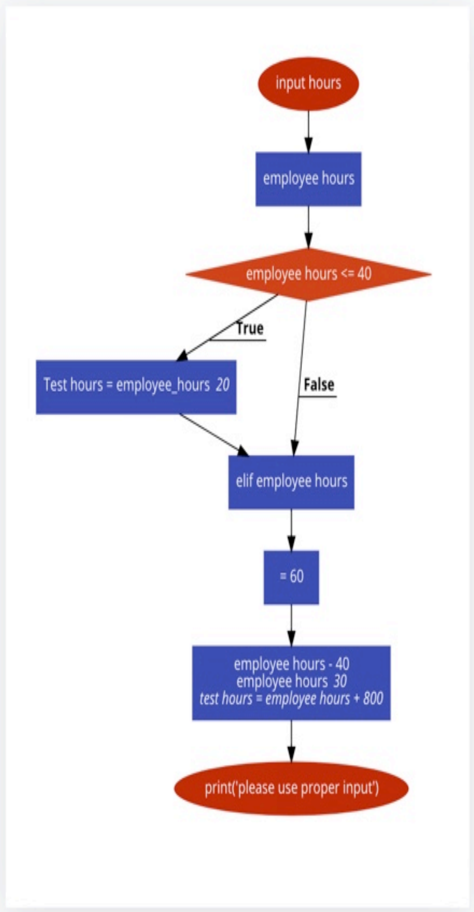
The best types of programs are simple, and if they cannot be simple than they should be simplified for multiple variations of people to use.



```
input hours;
employee hours;
if(employee hours <= 40) {
  Test hours = employee_hours * 20
}

elif employee hours <= 60 {
  employee hours - 40
  employee hours * 30
  test hours = employee hours + 800
}
Else:
    print('please use proper input')

```


input hours
employee hours
employee hours <= 40
True
False
Test hours = employee_hours 20
elif employee hours
= 60
employee hours - 40
employee hours 30
test hours = employee hours + 800
print('please use proper input')