Problem Set – Introduction to Functions.

1. Allow the user to repeatedly enter a quantity and price. Prompt the user on whether they want to do the program (Yes or No). Use a function to compute the total (quantity times price). The function should be passed the quantity and price and then return the total. In the function, provide a 10% discount if the total is over $10,0000.00. Display quantity, price and total. Sum and display the extended price.

|  |  |  |
| --- | --- | --- |
| Input | process | output |
| Qty  price | Function for qty \*price  Return total | Qty  Price  total |
| Yes or no | 10% discount if total >10,000 | Ext price |
|  | def compextprice(qty, unitprice):  extprice=qty\*unitprice  if extprice >10000:  discamt=extprice\* .10  else:  discamt=0    extprice=extprice-discamt  return extprice |  |
|  |  |  |

1. Enter players last name, number of hits and at bats at the keyboard. Prompt the user on whether they want to do the program (Yes or No). Use a function to compute batting average. Pass the hits and at bats to the function. The function should return batting average. Display last name and batting average. Give a count of the number of players entered.

|  |  |  |
| --- | --- | --- |
| input | process | output |
| lname | avgFunction=bathit/atbat | lname |
| Numberofh  At bat | Loop= inputs and outputs and avg func | Batting avg |
| Do you want to run program | def compbatavg (hit,atbat):  batavg=hit/atbat  return batavg | count |
|  |  |  |

1. Enter the destination city, miles travelled and gallons used for a trip. Prompt the user on whether they want to do the program (Yes or No). Use a function to compute miles per gallon. Pass miles travelled and gallons used to the function. The function should return miles per gallon. Count the number of entries made (number of trips) Display destination city, miles and mpg. At end display the number of entries made.

|  |  |  |
| --- | --- | --- |
| input | process | output |
| City  Milest | Mpgfucntion=milest/gall | Number of entries made/trips made |
| gall | def commpg (milest,gall):  mpg=milest/gall  return mpg | Destination city  Miles  mpg |
| Yes or no |  |  |
|  |  |  |

1. Allow the employee to enter last name, job code and hours worked. Prompt the user on whether they want to do the program (Yes or No). Use a function to determine the pay rate. Pass to this function the job code and it should return rate of pay. Use Job code L is $25/hr, A is $30/hr and J is $50/hr for respective pay rates. Compute gross pay. Give time and a half for overtime. Display last name and gross pay. Sum and display total of all gross pay.

|  |  |  |
| --- | --- | --- |
| input | process | output |
| lname | Function= def compayrate (jobc):  if jobc=='L':  prate=25  elif jobc=='A':  prate=30  else:  prate=50        return prate | lname |
| Jobc(L=$25,A=$30,J=$50) | Grosspay=payrate\*hrs | Gross pay |
| hours |  | All gross pay |
| Yes or no |  |  |

1. Allow the user to enter student last name, credit hours and district code. Prompt the user on whether they want to do the program (Yes or No). Use a function to compute tuition owed. Charge In district (code of I) $250 per credit hour. Out of district (code of O) is $550 per credit hour. The function should receive credit hours and district code and return tuition owed. Display student name and tuition owed. Sum and display total of all tuition owed.

|  |  |  |
| --- | --- | --- |
| input | process | output |
| lname | def comtuition(credhrs,discode):  if discode=='I':  return credhrs\*250  elif discode=='O':  return credhrs\*550 | Lname |
| Credit hrs |  | Tution owed |
| Discoed(I=$250,O=$550 |  |  |
|  |  |  |

What value do functions provide to your problem solutions? Functions seem to add more code to the program but what are the advantages of this code?

Functions add the ability to call back to a certain action which allows for quick and easy use of that said action knowing it will be work 100% of the time. It can be used multiple times, and it makes for an easier time when coding on larger and more complex projects. Troubleshooting is made much simpler knowing that its not a simple mistake like the specific function that you made. While functions do add more code in the long-term use of it during a large project it will save you the hassle of rewriting and remembering code. This reduces the chances of making errors and cuts down on redundant code. Having specific functions outside of the main code tidy’s up the entire thing and it makes for easy access when looking for them. By focusing on specific functions, developers can better grasp how each part of the program works. Organizing code into functions results in cleaner, more readable code, which helps developers or collaborators quickly understand the structure of the program. In summary, functions streamline coding, making development faster, debugging easier, and the code more maintainable and scalable in the long run. It’s a great tool for both beginners and experienced developers that can’t be replaced due to its efficiency boost and reliability. Code is at its best when made efficient and that’s is why this is a crucial part of writing good code.