Problem Set – More on Pass By Value Functions. Create an IPO for each of the problems below. Save the document with the IPO’s and then upload to Blackboard. Next write code for the problems. Then upload the .cpp files to Blackboard.

1. Allow the user to enter a **quantity** and **price**, use ctl+z to stop. Use a ***function*** to compute the **total** (quantity times price). The function should be passed the quantity and price and then return the total. Use another ***function*** to compute 10% discount if the total is over $10,000.00 and 5% for any amount equal to or lower than $10,000.00. The second function should receive the total, check which discount rate to charge, compute discount amount and then compute the **discount total** (total – discount amount). It should return the **discount total**. Display **total** and **discount total**. **Sum total** and **discount total** and display at the end.

|  |  |  |
| --- | --- | --- |
| Input | Process | Output |
| qty | function:compute\_total  input:qty, price  total=qty\*price  return total | inside loop  total  discounttotal |
| price | function:compute\_discounttotal  input:total  if(total>10000)  discount=total\*.1  else  discount=total\*.05  discounttotal=total-discount  return discounttotal | outside loop  sumtotal  sumdiscounttotal |
|  | sumtotal=0  sumdiscounttotal=0 |  |
|  | while not eof |  |
|  | total=compute\_total(qty,price) |  |
|  | discounttotal=compute\_discount(total) |  |
|  | sumtotal=sumtotal+total  sumdiscounttotal=sumdiscounttotal+discounttotal |  |
|  | Display:total, discounttotal  ask prompt again |  |
|  | end of loop |  |

1. Enter players **last name**, number of **hits** and at **bats** at the keyboard, use ctl+z to stop. Use a ***function*** to compute **batting average**. Pass the hits and at bats to the function. The function should return batting average (at bats / number of hits). Display **last name** and **batting average**. Give a count of the number of players entered and display the count after the loop.

|  |  |  |
| --- | --- | --- |
| Input | Process | Output |
| lname | function:compute\_bataverage  input:numbats, numhits  bataverage=numbat/numhits  return bataverage | inside loop  lname  bataverage |
| numhits | numcout=0 | outside loop  numcount |
| numbats | while not eof |  |
|  | bataverage=computebataverage(numhtits, numbats) |  |
|  | numcount=numcount+1 |  |
|  | display:lname, bataverage  ask prompt again |  |
|  | end of loop |  |

1. Enter the destination **city**, **miles** travelled and **gallons** used for a trip, use ctl+z to stop. Use a ***function*** to compute **miles per gallon**. Pass miles travelled and gallons used to the function. The function should return miles per gallon. Use another ***function*** to compute **gas cost**. Pass to this function gallons used. Each gallon costs $3.50. Compute and return the cost. Display destination **city**, **miles per gallon** and **cost of gas**. Sum and display the total cost of gas.

|  |  |  |
| --- | --- | --- |
| Input | Process | Output |
| city | function:compute\_mpg  input:miles, gallons  mpg=miles/gallons  return mpg | inside loop  city  mpg  gascost |
| miles | function:compute\_gascost  input:gallons  gascost=gallons\*3.5  return gascost | outside loop  sumgascost |
| gallons | sumgascost=0 |  |
|  | while not eof |  |
|  | mpg=compute\_mpg(miles,gallons) |  |
|  | gascost=compute\_gascost(gallons) |  |
|  | sumgascost=sumgascost+gascost |  |
|  | display:city, mpg, gascost  ask prompt again |  |
|  | end of loop |  |

1. Allow the employee to enter **last name, job code** and **hours** worked, use ctl+z to stop. Use a ***function*** to determine the **pay rate**. Pass to this function the job code and it should return rate of pay. Use the following rates based on Job code: L is $25/hr, A is $30/hr and J is $50/hr for respective pay rates. Write another ***function*** to determine the **gross pay**. Pass to this function the hours worked and pay rate and return 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:compute\_payrate  input:code  switch (code)  case ‘L’:  payrate=25  case ’A’:  payrate=30  case ‘J’:  payrate=50  default  incorrect jobcode  return payrate | inside loop  lname  grosspay |
| code | function:compute\_grosspay  input:hours, payrate  if(hours>40)  overtimehours=hours-40  grosspay=(payrate\*40)+(payrate\*1.5\*overtimehours)  else  grosspay=payrate\*hours  return grosspay | outside loop  sumgrosspay |
| hours | sumgrosspay=0 |  |
|  | while not eof |  |
|  | payrate=compute\_payrate(code) |  |
|  | grosspay=compute\_grosspay(hours, payrate) |  |
|  | sumgrosspay=sumgrosspay+grosspay |  |
|  | display:lname, grosspay  ask prompt again |  |
|  | end of loop |  |

1. Allow the user to enter student **last name**, **credit hours** and **district code**, use ctl+z to stop. Use a ***function*** to compute **tuition** owed. First, write a function to determine the cost per credit hour. Charge In district (code of I) $250 per credit hour. Out of district (code of O) is $550 per credit hour. Write another **function** to compute **tuition cost**. Display student **name** **and tuition cost**. Sum and display total of all tuition costs.

|  |  |  |
| --- | --- | --- |
| Input | Process | Output |
| lname | function:compute\_tuitionrate  input:code  switch(code)  case ‘I’:  tuitionrate=250  case ‘O’:  tuitionrate=550  default:  incorrect code  return tuitionrate | inside loop  lname  tuitioncost |
| credithours | function:tuitioncost  input:credithours, tuitionrate  tuitioncost=credithours\*tuitionrate  return tuitioncost | outside loop  sumtuitioncost |
| code | sumtuitioncost=0 |  |
|  | while not eof |  |
|  | tuitionrate=compute\_tuitionrate(code) |  |
|  | tuitioncost=compute\_tuitioncost(credithours, tuitionrate) |  |
|  | sumtuitioncost=sumtuitioncost+tuitioncost |  |
|  | display:lname, tuitioncost  ask prompt again |  |
|  | end of loop |  |