Jun Kim

Satish Singhal

CSCS-1 #0105

Apr. 30, 2021

Test 2: IOAA Document

**Data Input**

|  |  |  |
| --- | --- | --- |
| Variable name | C++ Data Type | Remarks/Comments |
| first | string | first name of the customer |
| last | string | last name of the customer |
| size | char | Code for sandwich size, refer to Table 1 (Upper case letter ONLY) |
| flourType | int | Type of flour used to make the sandwich bread, refer to Table 2 |
| fillingType | int | Filling placed inside the sandwich, refer to Table 3 |

Table 1. Sandwich size

|  |  |  |
| --- | --- | --- |
| Sandwich Size | Sandwich size code that user will enter [Upper case letter ONLY] | Base Cost ($) |
| Six Inch | S | 1.03 |
| Twelve Inch | T | 2.11 |
| Eighteen Inch | E | 3.39 |
| Two Feet | F | 4.11 |

Table 2. Flour type

|  |  |  |
| --- | --- | --- |
| Flour Type | Flour Code | Extra Cost |
| White | 1 | No Extra cost |
| Organic Wheat | 2 | Add 10.11% of the base cost |
| Multigrain | 3 | Add 15.55% of the base cost |
| Organic Multigrain | 4 | Add 20.12% of the base cost |

Table 3. Filling type

|  |  |  |
| --- | --- | --- |
| Filling Name | Filling Code | Extra Cost |
| Cheese | 1 | No Extra Cost |
| Artichokes | 2 | 0.50 + 10.1% of base cost |
| Jalapenos | 3 | 0.25 + 15.5% of base cost |
| Extra Cheese | 4 | 0.30 + 11.3% of base cost |
| All of above | 5 | Add all of above costs |

**Data Output**

|  |  |  |
| --- | --- | --- |
| Variable name | C++ Data Type | Remarks/Comments |
| first | string | first name of the customer |
| last | string | last name of the customer |
| dollars | int | Whole dollars portion of the cost |
| cents | int | Whole cents portion of the cost |

**Computational Aid and Other Variables**

|  |  |  |
| --- | --- | --- |
| Variable name | C++ Data Type | Remarks/Comments |
| flagSize | bool | Set to true, if size is ‘S’, ‘T’, ‘E’, or ‘F’ |
| flagFlour | bool | Set to true, if flourType is 1, 2, 3, or 4 |
| flagFilling | bool | Set to true, if fillingType is 1, 2, 3, 4, or 5 |
| baseCost | double | Base cost of the sandwich |
| incrFlour | double | Extra cost due to flour type |
| incrFilling | double | Extra cost due to fillings |
| totalCost | double | Total raw cost including tax |
| totalPennies | int | Final cost in rounded pennies |

**Global Constants**

|  |  |  |  |
| --- | --- | --- | --- |
| Variable name | C++ Data Type | Value | Remark |
| SIZE\_S | double | 1.03 | Cost in dollars for Six Inch |
| SIZE\_T | double | 2.11 | Cost in dollars for Twelve Inch |
| SIZE\_E | double | 3.39 | Cost in dollars for Eighteen Inch |
| SIZE\_F | double | 4.11 | Cost in dollars for Two Feet |
| FLOUR\_ORG\_WHEAT | double | 0.1011 | Fraction of base cost added if the sandwich bread is made of Organic Wheat |
| FLOUR\_MULTI | double | 0.1555 | Fraction of base cost added if the sandwich bread is made of Multigrain |
| FLOUR\_ORG\_MULTI | double | 0.2012 | Fraction of base cost added if the sandwich bread is mad of Organic Multigrain |
| FILLING\_ARTICHOKES | double | 0.50 | Cost in dollars added for Artichokes filling |
| FILLING\_ARTICHOKES\_PCT\_BASE | double | 0.101 | Fraction of base cost added for Artichokes filling |
| FILLING\_JALAPENOS | double | 0.25 | Cost in dollars added for Jalapenos filling |
| FILLING\_JALAPENOS\_PCT\_BASE | double | 0.155 | Fraction of base cost added for Jalapenos filling |
| FILLING\_EXTRA\_CHEESE | double | 0.30 | Cost in dollars added for Extra Cheese filling |
| FILLING\_EXTRA\_CHEESE\_PCT\_BASE | double | 0.113 | Fraction of base cost added for Extra Cheese filling |
| SALES\_TAX | double | 0.101 | Sales Tax (10.1%) |

**Analysis**

|  |  |
| --- | --- |
| C++ Computational Expression | Comments |
| incrFlour = baseCost \* FlourMultiplier | FlourMultipier is all the constants starting with words FLOUR |
| incrFilling = addedCost + (baseCost \* FillingMultipier) | addedCost is all the constants starting with words FILLING, and FillingMultipier is constants starting with word FILLING and ending with words PCT\_BASE |
| totalCost = (baseCost + incrFlour + incrFilling) \* (1+SALES\_TAX) | Total cost in dollars after adding tax |
| totalPennis = static\_cast<int>((totalCost \* 100)+0.5) | Cost in rounded pennies |
| dollars = totalPennis / 100 | Whole dollars portion of the cost |
| cents = totalPennis % 100 | Whole cents portion of the cost |

\* Figure 1, Checking Flags for inputs

flagSize = (size == ‘S’ or size == ‘T’ or size == ‘E’ or size == ‘F)

flagFlour = (flourType == 1 or flourType == 2 or flourType == 3 or flourType == 4)

flagFilling = (fillingType == 1 or fillingType == 2 or fillingType == 3 or fillingType == 4 or fillingType ==5)

\*Figure 2, Checking input values

If (flagSize and flagFlour and flagFilling) then

If (size == ‘S’) then

baseCost = SIZE\_S

Else if (size == ‘T’) then

baseCost = SIZE\_T

Else if (size == ‘E’) then

baseCost = SIZE\_E

Else

baseCost = SIZE\_F

End if

If (flourType == 1) then

incrFlour = 0

Else if (flourType == 2) then

incrFlour = baseCost \* FLOUR\_ORG\_WHEAT

Else if (flourType == 3) then

incrFlour = baseCost \* FLOUR\_MULTI

Else

incrFlour = baseCost \* FLOUR\_ORG\_MULTI

End if

If (fillingType == 1) then

incrFilling = 0

Else if (fillingType == 2) then

incrFilling = FILLING\_ARTICHOKES + (baseCost \*

FILLING\_ARTICHOKES\_PCT\_BASE)

Else if (fillingType == 3) then

incrFilling = FILLING\_JALAPENOS + (baseCost \*

FILLING\_JALAPENOS\_PCT\_BASE)

Else if (fillingType == 4) then

incrFilling = FILLING\_EXTRA\_CHEESE + (baseCost \*

FILLING\_EXTRA\_CHEESE\_PCT\_BASE)

Else

incrFilling = FILLING\_ARTICHOKES + (baseCost \*

FILLING\_ARTICHOKES\_PCT\_BASE) +

FILLING\_JALAPENOS + (baseCost \*

FILLING\_JALAPENOS\_PCT\_BASE) +

FILLING\_EXTRA\_CHEESE + (baseCost \*

FILLING\_EXTRA\_CHEESE\_PCT\_BASE)

End if

\* Figure 3, Error message for flags

Else

Print, “We are unable to process your order due to following reasons.”, EOL

If (!flagSize) then

Print, “Invalid sandwich size was specified.”, EOL

End if

If (!flagFlour) then

Print, “Invalid flour code was entered.”, EOL

End if

If (!flagFilling) then

Print, “Invalid filling code was entered.”, EOL

End if

End if

**Algorithm**

1. Add all the #include directives, declare global constants
2. declare first as string
3. declare last as string
4. Print asking user first and last name, EOL
5. prompt user for first name
6. get and store input into variable first
7. prompt user for last name
8. get and store input into variable last
9. Print greeting message, first, last, EOL
10. Print the menu of sandwich size
11. declare size as int
12. prompt to enter sandwich size (‘S’, ‘T’, ‘E’, ‘F’)
13. get and store input into variable size
14. set flagSize for sandwich size as below

flagSize = (size == ’S’ || size == ‘T’ || size == ‘E’ || size = ‘F’)

1. Print the menu of flour type
2. declare flourType as int
3. prompt to enter flour type (1, 2, 3, 4)
4. get and store input into variable flourType
5. get flagFlour for flour type of sandwich bread as below

flagFlour = (flourType == 1 || flourType == 2 || flourType == 3 || flourType == 4)

1. Print the menu of filling type
2. declare fillingType as int
3. prompt to enter filling type (1, 2, 3, 4, 5)
4. get and store input into variable fillingType
5. get flagFilling for filling type into sandwich as below

flagFilling = (fillingType == 1 || fillingType == 2 || fillingType == 3 || fillingType == 4 || fillingType == 5)

1. If (flagSize && flagFlour && flagFilling) then
   1. declare baseCost as double
   2. declare incrFlour as double
   3. declare incrFilling as double
   4. If (size == ‘S’) then
      1. baseCost = SIZE\_S
   5. Else if (size == ‘T’) then
      1. baseCost = SIZE\_T
   6. Else if (size == ‘E’) then
      1. baseCost = SIZE\_E
   7. Else
      1. baseCost = SIZE\_F
   8. End if
   9. If (flourType == 1) then
      1. incrFlour = 0
   10. Else if (flourType == 2) then
       1. incrFlour = baseCost \* FLOUR\_ORG\_WHEAT
   11. Else if (flourType == 3) then
       1. incrFlour = baseCost \* FLOUR\_MULTI
   12. Else
       1. incrFlour = baseCost \* FLOUR\_ORG\_MULTI
   13. End if
   14. If (fillingType == 1) then
       1. incrFilling = 0
   15. Else if (fillingType == 2) then
       1. incrFilling = FILLING\_ARTICHOKES +

(baseCost \* FILLING\_ARTICHOKES\_PCT\_BASE)

* 1. Else if (fillingType == 3) then
     1. incrFilling = FILLING\_JALAPENOS +

(baseCost \* FILLING\_JALAPENOS\_PCT\_BASE)

* 1. Else if (fillingType == 4) then
     1. incrFilling = FILLING\_EXTRA\_CHEESE +

(baseCost \* FILLING\_EXTRA\_CHEESE\_PCT\_BASE)

* 1. Else
     1. incrFilling = FILLING\_ARTICHOKES +

(baseCost \* FILLING\_ARTICHOKES\_PCT\_BASE) +

FILLING\_JALAPENOS +

(baseCost \* FILLING\_JALAPENOS\_PCT\_BASE) +

FILLING\_EXTRA\_CHEESE +

(baseCost \* FILLING\_EXTRA\_CHEESE\_PCT\_BASE)

* 1. End if
  2. declare totalCost as double
  3. declare totalPennies as int
  4. declare dollars as int
  5. declare cents as int
  6. totalCost = (baseCost + incrFlour + incrFilling) \* (1 + SALES\_TAX)
  7. totalPennies = static\_cast<int>((totalCost \* 100) + 0.5)
  8. dollars = totalPennies / 100
  9. cents = totalPennies % 100
  10. set outputted formatted to two decimal places
  11. Print, dollars, “ dollars and ”,cents,” cents.”, EOL

1. Else
   1. Print error message, one or more codes were invalid., EOL
   2. If (!flagSize) then
      1. Print, “Invalid sandwich size was specified.”, EOL
   3. End if
   4. If (!flagFlour) then
      1. Print, “Invalid flour code was entered.”, EOL
   5. End if
   6. If (!flagFilling) then
      1. Print, “Invalid filling code was entered.”, EOL
   7. End if
2. End if