M. Hellard, J. Laframboise, C. Miller
B. Patrick
COIS 2020H
November 13th 2022

Assignment 2 - Testing

PART A:

Scenario 1: invalid menu selection

```
Open Hashtable Revisited
Note: for this demo, all hashtables store ints, and all keys are Points

Menu:
Empty HT               (c):
Insert point into HT   (i):
Remove point from HT   (d):
Retrieve point from HT (r):
Print entire HT        (p):
Make, print dev HT     (a):
Quit                   (q):
Input :Z
Unrecognised input, please try again

Menu:
Empty HT               (c):
Insert point into HT   (i):
Remove point from HT   (d):
Retrieve point from HT (r):
Print entire HT        (p):
Make, print dev HT     (a):
Quit                   (q):
Input :
```

I - Inserting point into HT

```
Open Hashtable Revisited
Note: for this demo, all hashtables store ints, and all keys are Points

Menu:
Empty HT               (c):
Insert point into HT   (i):
Remove point from HT   (d):
Retrieve point from HT (r):
Print entire HT        (p):
Make, print dev HT     (a):
Quit                   (q):
Input :i
```

Scenario I-1: Valid numerical input

```
Open Hashtable Revisited
Note: for this demo, all hashtables store ints, and all keys are Points

Menu:
Empty HT               (c):
Insert point into HT   (i):
Remove point from HT   (d):
Retrieve point from HT (r):
Print entire HT        (p):
Make, print dev HT     (a):
Quit                   (q):
Input :i

Input key x (int):2

Input key y (int):6

Input value (int):8

Menu:
Empty HT               (c):
Insert point into HT   (i):
Remove point from HT   (d):
Retrieve point from HT (r):
Print entire HT        (p):
Make, print dev HT     (a):
Quit                   (q):
Input :
```

Scenario I-2: Non-numerical input

```
Open Hashtable Revisited
Note: for this demo, all hashtables store ints, and all keys are Points

Menu:
Empty HT               (c):
Insert point into HT   (i):
Remove point from HT   (d):
Retrieve point from HT (r):
Print entire HT        (p):
Make, print dev HT     (a):
Quit                   (q):
Input :i

Input key x (int):a
Input error, please try again

Input key x (int):
```

II - Removing point from HT

```
Open Hashtable Revisited
Note: for this demo, all hashtables store ints, and all keys are Points

Menu:
Empty HT               (c):
Insert point into HT   (i):
Remove point from HT   (d):
Retrieve point from HT (r):
Print entire HT        (p):
Make, print dev HT     (a):
Quit                   (q):
Input :i

Input key x (int):2

Input key y (int):6

Input value (int):8

Menu:
Empty HT               (c):
Insert point into HT   (i):
Remove point from HT   (d):
Retrieve point from HT (r):
Print entire HT        (p):
Make, print dev HT     (a):
Quit                   (q):
Input :d
```

Scenario II-1: Valid numerical input

```
Input key x (int):2

Input key y (int):6

Input value (int):8

Menu:
Empty HT               (c):
Insert point into HT   (i):
Remove point from HT   (d):
Retrieve point from HT (r):
Print entire HT        (p):
Make, print dev HT     (a):
Quit                   (q):
Input :d

Input key x (int):2

Input key y (int):6
Successfully removed

Menu:
Empty HT               (c):
Insert point into HT   (i):
Remove point from HT   (d):
Retrieve point from HT (r):
Print entire HT        (p):
Make, print dev HT     (a):
Quit                   (q):
Input :
```

Scenario II-2: Non-numerical input

```
Open Hashtable Revisited
Note: for this demo, all hashtables store ints, and all keys are Points

Menu:
Empty HT               (c):
Insert point into HT   (i):
Remove point from HT   (d):
Retrieve point from HT (r):
Print entire HT        (p):
Make, print dev HT     (a):
Quit                   (q):
Input :d

Input key x (int):a
Input error, please try again

Input key x (int):
```

III - Retrieving point from HT

```
Open Hashtable Revisited
Note: for this demo, all hashtables store ints, and all keys are Points

Menu:
Empty HT               (c):
Insert point into HT   (i):
Remove point from HT   (d):
Retrieve point from HT (r):
Print entire HT        (p):
Make, print dev HT     (a):
Quit                   (q):
Input :i

Input key x (int):2

Input key y (int):6

Input value (int):8

Menu:
Empty HT               (c):
Insert point into HT   (i):
Remove point from HT   (d):
Retrieve point from HT (r):
Print entire HT        (p):
Make, print dev HT     (a):
Quit                   (q):
Input :r
```

Scenario III-1: Valid numerical input

```
Input key x (int):2

Input key y (int):6

Input value (int):8

Menu:
Empty HT               (c):
Insert point into HT   (i):
Remove point from HT   (d):
Retrieve point from HT (r):
Print entire HT        (p):
Make, print dev HT     (a):
Quit                   (q):
Input :r

Input key x (int):2

Input key y (int):6
Retrieved value :8

Menu:
Empty HT               (c):
Insert point into HT   (i):
Remove point from HT   (d):
Retrieve point from HT (r):
Print entire HT        (p):
Make, print dev HT     (a):
Quit                   (q):
Input :
```

Scenario III-2: Invalid numerical input

```
Open Hashtable Revisited
Note: for this demo, all hashtables store ints, and all keys are Points

Menu:
Empty HT               (c):
Insert point into HT   (i):
Remove point from HT   (d):
Retrieve point from HT (r):
Print entire HT        (p):
Make, print dev HT     (a):
Quit                   (q):
Input :r

Input key x (int):55

Input key y (int):66
Key not found

Menu:
Empty HT               (c):
Insert point into HT   (i):
Remove point from HT   (d):
Retrieve point from HT (r):
Print entire HT        (p):
Make, print dev HT     (a):
Quit                   (q):
Input :
```

Scenario III-3: Non-numerical input

```
Menu:
Empty HT               (c):
Insert point into HT   (i):
Remove point from HT   (d):
Retrieve point from HT (r):
Print entire HT        (p):
Make, print dev HT     (a):
Quit                   (q):
Input :i

Input key x (int):2

Input key y (int):6

Input value (int):8

Menu:
Empty HT               (c):
Insert point into HT   (i):
Remove point from HT   (d):
Retrieve point from HT (r):
Print entire HT        (p):
Make, print dev HT     (a):
Quit                   (q):
Input :r

Input key x (int):e
Input error, please try again

Input key x (int):
```

IV - Printing entire HT

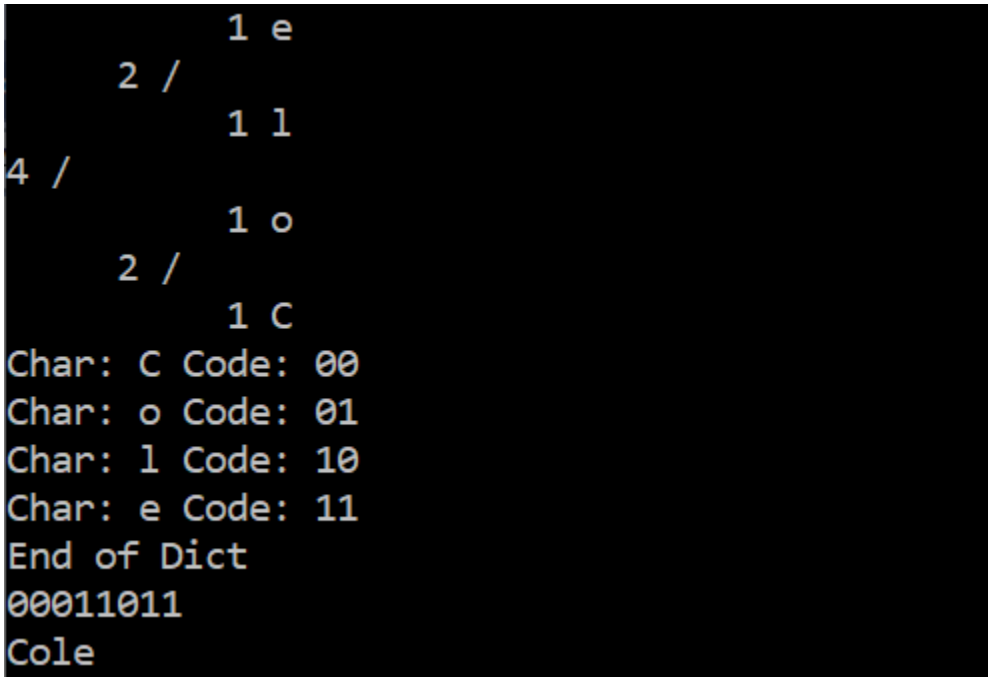
```
Note: for this demo, all hashtables store ints, and all keys are Points

Menu:
Empty HT               (c):
Insert point into HT   (i):
Remove point from HT   (d):
Retrieve point from HT (r):
Print entire HT        (p):
Make, print dev HT     (a):
Quit                   (q):
Input :i

Input key x (int):2

Input key y (int):6

Input value (int):8

Menu:
Empty HT               (c):
Insert point into HT   (i):
Remove point from HT   (d):
Retrieve point from HT (r):
Print entire HT        (p):
Make, print dev HT     (a):
Quit                   (q):
Input :p
 0: <[2 6],8>
hit enter to return
```

V - Making, printing dev HT

```
Input :a
 0: <[9 0],91> <[3 0],31> <[2 0],21> <[0 5],6> <[0 4],5> <[0 3],4> <[0 2],3> <[0 1],2> <[0 0],1> <[0 6],7> <[0 7],8> <[0 8],9> <[0 9],10> <[1 0],11> <[4 0],41> <[5 0],51> <
[6 0],61> <[7 0],71> <[8 0],81>
 1: <[1 1],12>
 2: <[2 1],22> <[1 2],13>
 3: <[3 1],32> <[1 3],14> <[5 8],59> <[8 5],86>
 4: <[2 2],23> <[1 4],15> <[4 1],42>
 5: <[1 5],16> <[5 1],52> <[6 7],68> <[7 6],77>
 6: <[3 2],33> <[2 3],24> <[1 6],17> <[6 1],62>
 7: <[1 7],18> <[7 1],72> <[9 9],100>
 8: <[2 4],25> <[1 8],19> <[4 2],43> <[8 1],82> <[9 5],96> <[5 9],60>
 9: <[9 1],92> <[3 3],34> <[1 9],20>
10: <[2 5],26> <[5 2],53>
11: <[8 6],87> <[6 8],69>
12: <[3 4],35> <[2 6],27> <[4 3],44> <[6 2],63> <[7 7],78>
13:
14: <[2 7],28> <[7 2],73>
15: <[3 5],36> <[5 3],54>
16: <[2 8],29> <[4 4],45> <[8 2],83>
17: <[9 6],97> <[6 9],70>
18: <[9 2],93> <[2 9],30> <[3 6],37> <[6 3],64>
19: <[8 7],88> <[7 8],79>
20: <[4 5],46> <[5 4],55>
21: <[3 7],38> <[7 3],74>
22:
23:
24: <[3 8],39> <[4 6],47> <[6 4],65> <[8 3],84>
25: <[5 5],56>
26: <[9 7],98> <[7 9],80>
27: <[9 3],94> <[3 9],40> <[8 8],89>
28: <[4 7],48> <[7 4],75>
29:
30: <[5 6],57> <[6 5],66>
31:
32: <[4 8],49> <[8 4],85>
33:
34:
35: <[5 7],58> <[7 5],76> <[9 8],99> <[8 9],90>
36: <[9 4],95> <[4 9],50> <[6 6],67>
hit enter to return
```

PART B:

Scenario 1: Frequency of every character is the same



Scenario 2: Sentence with varying frequencies.

```
                    2 i
               4 /
                    2 g
          7 /
               3 o
     11 /
               2
          4 /
                    1 a
               2 /
                    1 P
19 /
                    1 l
               2 /
                    1 s
          4 /
               2 m
     8 /
                    1 c
               2 /
                    1 n
          4 /
               2 r
Char: r Code: 000
Char: n Code: 0010
Char: c Code: 0011
Char: m Code: 010
Char: s Code: 0110
Char: l Code: 0111
Char: P Code: 1000
Char: a Code: 1001
Char:   Code: 101
Char: o Code: 110
Char: g Code: 1110
Char: i Code: 1111
End of Dict
10000001101110000100101001011110010111010111110110101001111011001111
Programming is cool
```