CS1342 – Spring 2023

Program 1

In this program, you utilize C++ fundamentals (Console I/O, Strings, Branches, Loops and Operators) to encode and decode integers and string values. Your program must be submitted to GitHub by 6:00 AM CST on **Monday, February 6, 2023.** You will receive a 30 point penalty for any program submitted within 48 hours after the due date. Any programs submitted after 6:00 AM CST on **Wednesday, February 7, 2023** will receive a 0.

Overview

Most manufactured products ship with a UPC (Universal Product Code) number (and corresponding bar code) imprinted on the product packaging. Because not all products have a UPC, many retailers must create their own product identifier for each item in their inventory. This identifier is referred to as a SKU (stock keeping unit) number. For this program, you are supporting the owner of a new retail who needs software to generate a unique SKU for new products that are added to its inventory. This program must perform the following three functions:

1. Generate a 5-digit hash value generated from a string containing the product description.
2. Create a 6-digit product SKU by calculating and appending a check digit.
3. Verify a SKU by validating that the check digit is correct for the product hash value.

Program specifications:

You are to create a menu-driven program that will allow a user to generate a product hash value, generate a SKU, verify a SKU, or exit the system.

```
MENU
1. Generate product hash value
2. Generate SKU from product hash value
3. Verify SKU
9. Exit
Enter choice:
```

If a user enters any decimal value other than 1-3 or 9, print an error message and re- prompt the user for a correct entry. **Hint**: After using *cin* to prompt the user for data entry, call *cin.clear()* to flush out the input stream buffer.

Menu Option 1 – Generate Product Hash Value

Generate a numeric **product hash value** derived by converting a text description of the product into a numeric value. This algorithm should do the following:

- Prompt the user to enter a product description, captured as a string using the getline function.
- Convert each character in the product description to its ASCII decimal equivalent value.
- Sum these numeric ASCII values for all the characters in the description.

- Ensure that the **product hash value** has exactly 5 digits. If the sum of the ASCII values has fewer than 5 digits, add 0's to the end of the number as needed. You can assume that a product description will always have less than 80 characters and will never generate a number with more than 5 digits.

  Example:

  Product description: "med red SMU hat "

  ASCII conversion:

char	m	e	d		r	e	d		S	M	U		h	a	t	**TOTAL**
ASCII	109	101	100	32	114	101	100	32	83	77	85	32	104	97	116	**1283**

  Product hash value: 12830 (zero added to the end to adjust to fixed size of five digits)

  Hint:

```
//Function to count digits in a number
int countDigits(int num) {
    int count = 0;
    while(num) {
        count++;
        num /= 10;
    }
    return count;
}
```

  Note: This algorithm could potentially generate a duplicate hash value for two products this descriptions that contain the same characters. This would have to be reconciled against a list of previously assigned values, which is outside the scope of this program.

Menu Option 2 – Generate SKU

Generate a **SKU** from a **product hash value** by adding a check digit to the end, forming a 6-digit number where the first five digits are the product hash value and the last digit is the check digit. Check digits are commonly used in systems that involve data entry. When a SKU is entered by a user, the system can use the check digit to verify that rest of the digits were entered properly. The check digit is calculated using the following algorithm using product hash value **12830** as an example:

- Add all of the digits in the odd positions (digits in position 1, 3, and 5) **1 + 8 + 0 = 9**
- Multiply by 3. **9 * 3 = 27**
- Add all of the digits in even positions (digits in position 2 and 4). **2 + 3 = 5**
- Sum the results of steps 3 and 2. **27 + 5 = 32**
- Determine what number needs to be added to the result of the previous step n order to create an even multiple of 10. **32 + 8 = 40 (8 is the check digit)**
- This check digit is placed at the end of the product hash value to form the 6-digit SKU: **128308**

Option 3 – Verify SKU

Verify a SKU by validating that the check digit is correct for the product hash value.

Enter SKU: **128308**

Result: ***Valid SKU***

Enter SKU: **128301**

Result: ***Invalid SKU***

Option 4 – Exit

If a user enters 9 your program should end.

Sample Output

```
MENU
1. Generate product hash value
2. Generate SKU from product hash value
3. Verify SKU
9. Exit
Enter choice: 1
Enter description: med red SMU hat
Product number: 12830

MENU
1. Generate product hash value
2. Generate SKU from product hash value
3. Verify SKU
9. Exit
Enter choice: 2

Enter 5-digit product hash value: 12830
SKU with check digit = 128308

MENU
1. Generate product hash value
2. Generate SKU from product hash value
3. Verify SKU
9. Exit
Enter choice: 3
Enter 6-digit SKU: 128308
SKU is valid

MENU
1. Generate product hash value
2. Generate SKU from product hash value
3. Verify SKU
9. Exit
Enter choice: 3
Enter 6-digit SKU: 128304
SKU is not valid

MENU
1. Generate product hash value
2. Generate SKU from product hash value
3. Verify SKU
9. Exit
Enter choice: 9

Process finished with exit code 0
```