/\* -----------------------------------------------

Name: J-Zach Loke

Course: CMPS-385

Semester: Spring 2020

Project: No. 2 Part 1

Purpose: Read a positive integer number and convert it

to base 2, 8, and 16 using stack

----------------------------------------------- \*/

#include <iostream>

// our stack class

class Stack

{

private:

char contents[100];

int counter = 0;

public:

void clear() { counter = 0; }

bool isEmpty() { return (counter == 0) ? true : false; }

bool isFull() { return (counter == 100) ? true : false; }

void push(char c) { contents[counter++] = c; }

char pop() { return contents[--counter]; }

};

// function prototytpes

int getInput();

Stack changeBase(int base10, int newBase);

void displayStack(Stack src);

int main()

{

/\* name: main

input: N/A

output: N/A

purpose: main function to drive the program \*/

// variables

int input = getInput();

Stack base2 = changeBase(input, 2);

Stack base8 = changeBase(input, 8);

Stack base16 = changeBase(input, 16);

std::cout << "\t" << input << " at base 2 is\t";

displayStack(base2);

std::cout << "\t" << input << " at base 8 is\t";

displayStack(base8);

std::cout << "\t" << input << " at base 16 is\t";

displayStack(base16);

system("pause");

return 0;

}

int getInput()

{

/\* name: main

input: N/A

output: string input

purpose: prompts and returns a int entered by the user \*/

int input;

std::cout << "Enter a postiive int number: ";

std::cin >> input;

return input;

}

Stack changeBase(int base10, int newBase)

{

/\* name: changeBase

input: int base10

output: Stack converted

purpose: converts a base10 number to any base using stacks \*/

Stack converted;

while (base10 > 0)

{

int digit = base10 % newBase;

char fixedDigit = char('0' + digit);

if (digit >= 10) { fixedDigit += 7; }

converted.push(fixedDigit);

base10 /= newBase;

}

return converted;

}

void displayStack(Stack src)

{

/\* name: displayStack

input: string msg, Stack src

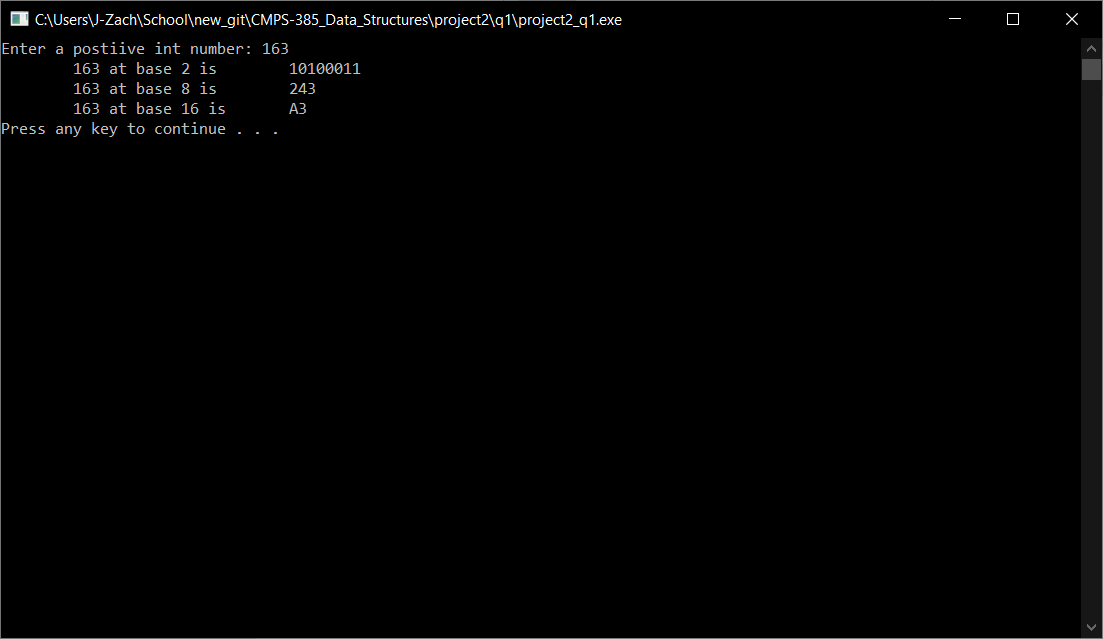
output: N/A

purpose: displays the contents of a stack without changing it \*/

while (!src.isEmpty()) { std::cout << src.pop(); }

std::cout << std::endl;

}



/\* -----------------------------------------------

Name: J-Zach Loke

Course: CMPS-385

Semester: Spring 2020

Project: No. 2 Part 2

Purpose: Store 10 random integers <100 in stack S.

(i) display all numbers

(ii) copy all even numbers in stack EVEN

and all odd numbers in stack ODD

(iii) display stack EVEN and ODD

----------------------------------------------- \*/

#include <iostream>

#include <time.h>

#include <string>

// our stack class

class Stack

{

private:

int contents[10];

int counter = 0;

public:

void clear() { counter = 0; }

bool isEmpty() { return (counter == 0) ? true : false; }

bool isFull() { return (counter == 10) ? true : false; }

void push(int n) { contents[counter++] = n; }

int pop() { return contents[--counter]; }

};

// function prototytpes

void fillStack(Stack& S);

void sortStacks(Stack src, Stack& even, Stack& odd);

void displayStack(std::string msg, Stack src);

int main()

{

/\* name: main

input: N/A

output: N/A

purpose: main function to drive the program \*/

// variables

Stack EVEN;

Stack ODD;

Stack S;

fillStack(S); // fill S with random integers < 100

displayStack("All Numbers:\t", S); // display all numbers

sortStacks(S, EVEN, ODD); // sort S contents into even and odd

displayStack("Even Numbers:\t", EVEN); // display all even numbers

displayStack("Odd Numbers:\t", ODD); // dispaly all odd numbers

system("pause");

return 0;

}

void fillStack(Stack& S)

{

/\* name: fillStack

input: Stack S

output: Stack S

purpose: fills the stack with 10 random integers < 100 \*/

srand(time(0));

for (int i = 0; i < 10; ++i)

{

S.push(rand()%100);

}

}

void sortStacks(Stack src, Stack& even, Stack& odd)

{

/\* name: sortStacks

input: Stack src, Stack even, Stack odd

output: Stack even, Stack odd

purpose: sorts src contents into even and odd \*/

while(!src.isEmpty())

{

int n = src.pop();

if (n % 2 == 0) { even.push(n); }

else { odd.push(n); }

}

}

void displayStack(std::string msg, Stack src)

{

/\* name: displayStack

input: string msg, Stack src

output: N/A

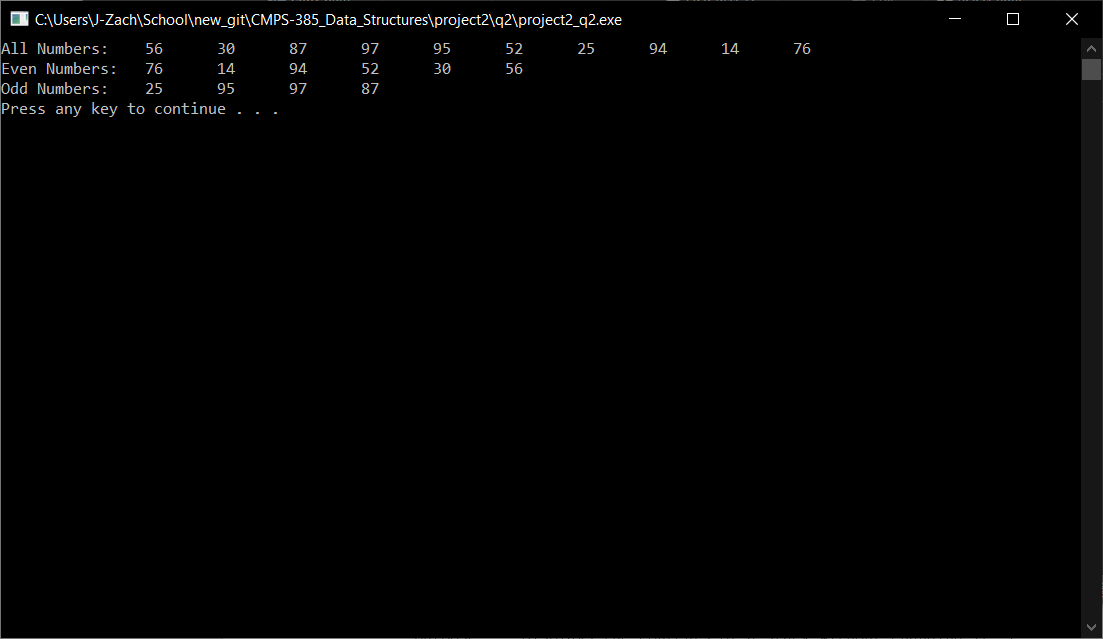
purpose: displays the contents of a stack without changing it \*/

std::cout << msg;

while (!src.isEmpty()) { std::cout << src.pop() << '\t'; }

std::cout << std::endl;

}



/\* -----------------------------------------------

Name: J-Zach Loke

Course: CMPS-385

Semester: Spring 2020

Project: No. 2 Part 3

Purpose: Read a sentence and display all lowercase letters,

uppercase letters, and vowels in reverse order as you read.

Use predefined fucntions: isupper, islower, and wirte your own

functions isVowel

----------------------------------------------- \*/

#include <iostream>

#include <string>

// our stack class

class Stack

{

private:

char contents[100];

int counter = 0;

public:

void clear() { counter = 0; }

bool isEmpty() { return (counter == 0) ? true : false; }

bool isFull() { return (counter == 100) ? true : false; }

void push(char c) { contents[counter++] = c; }

char pop() { return contents[--counter]; }

};

// function prototytpes

std::string getInput();

bool isVowel(char c);

void displayStack(std::string msg, Stack src);

int main()

{

/\* name: main

input: N/A

output: N/A

purpose: main function to drive the program \*/

// variables

Stack upper;

Stack lower;

Stack vowel;

std::string input = getInput();

// sort the string

for (std::string::size\_type i = 0; i < input.size(); ++i)

{

char c = input[i];

if (islower(c) && !lower.isFull()) { lower.push(c); }

if (isupper(c) && !upper.isFull()) { upper.push(c); }

if (isVowel(c) && !vowel.isFull()) { vowel.push(c); }

}

// output

displayStack("Uppercase letters:\t", upper);

displayStack("Lowercase letters:\t", lower);

displayStack("Vowels:\t\t\t", vowel);

system("pause");

return 0;

}

std::string getInput()

{

/\* name: main

input: N/A

output: string input

purpose: prompts and returns a sentence entered by the user \*/

std::string input;

std::cout << "Enter a sentence: ";

std::getline(std::cin, input);

return input;

}

bool isVowel(char c)

{

/\* name: isVowel

input: char c

output: bool

purpose: determines if a character is a vowel regardless of case \*/

c = tolower(c); // convert to lowercase

return (c == 'a' || c == 'e' || c == 'i' || c == 'o' || c == 'u') ? true : false;

}

void displayStack(std::string msg, Stack src)

{

/\* name: displayStack

input: string msg, Stack src

output: N/A

purpose: displays the contents of a stack without changing it \*/

std::cout << msg;

while (!src.isEmpty()) { std::cout << src.pop() << '\t'; }

std::cout << std::endl;

}

