Barak Barclay

Leslie Tekamp

ECE-1021

15 September 2015

HWK-2

1. 3.1 a) Algorithm. b) Program control. c) Sequence, selection, repetition. d) if…else. e) Compound statement or block. f) while. g) Counter-controlled or definite. h) Sentinel.
2. //

// HWK2(2).cpp

//

// By: Barak Barclay

// Date: 15 Sept 2015

//

// Problem Statement: Nested looping structures. Write a program that prompts the user to enter in a symbol. Read in the symbol. Next prompt the user to enter an integer for the size of a square box to print. Read in the integer. Use the symbol and size to print a square (filled-in) to the screen using a looping structure. The inner looping structure should only use a single printf() statement.

//

//

// %%%% Algorthim %%%%

//

// preprocessor directives

// start main

// init variables

// prompt user to enter symbol

// read in symbol

// prompt user to enter box size

// read in size

// while loop

// count1++

// print to next line

// set count2 = 0

// while loop

// count2++

// print symbol

// end while

// end while

// end main

//

#include <stdio.h>

int main(void)

{

char symbol = 0;

int size = 0;

int count1 = 1;

int count2 = 1;

printf("Enter a symbol:");

scanf\_s("%c", &symbol);

printf("\n\nEnter an integer for the size of a square box to print:");

scanf\_s("%d", &size);

printf("\n\n");

while (count1 <= size)

{

count1++;

printf("\n");

count2 = 0;

while (count2 < size)

{

count2++;

printf("%c", symbol);

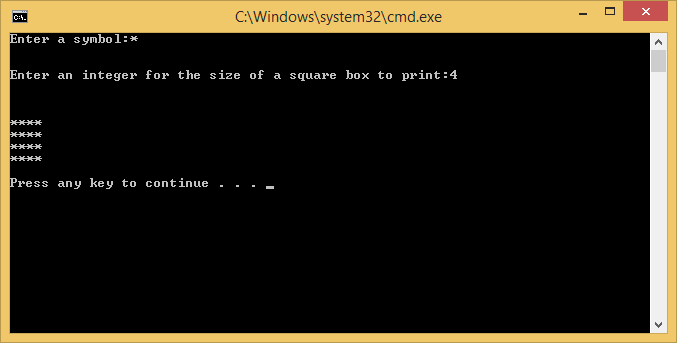
}

}

printf("\n\n");

return 0;

}



1. //

// HWK2(3).cpp

//

// By: Barak Barclay

// Date: 15 Sept 2015

//

// Problem Statement: Write a program that prompts the user to enter in a double number, read in the double number. Square the entered number and print the result to the monitor.

//

//

// %%%% Algorthim %%%%

//

// preprocessor directives

// start main

// init variables

// prompt user to enter double number

// read in number

// square number in squared varible

// print squared

// end main

//

#include <stdio.h>

int main(void)

{

double number = 0;

double squared = 0;

printf("Enter a double number:");

scanf\_s("%lf", &number);

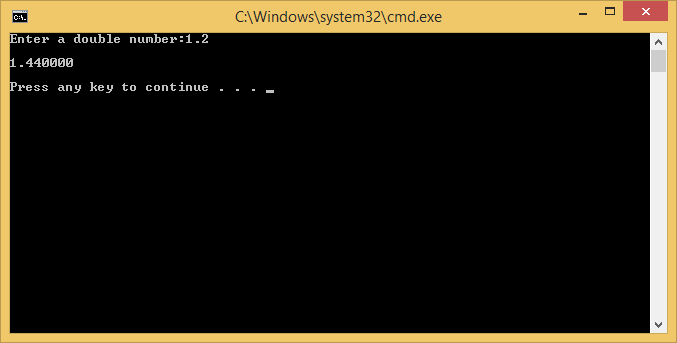
squared = number \* number;

printf("\n%lf", squared);

printf("\n\n");

return 0;

}



//

// HWK2(4).cpp

//

// By: Barak Barclay

// Date: 15 Sept 2015

//

// Problem Statement: Write a program that prompts the user to enter in a char data type called Grade from the set { ‘A’, ‘B’, ‘C’, ‘D’, ‘F’ }. Write a message to the monitor, which states you expect to get an (insert entered char) in this class. Assume the user will enter a valid Grade.

//

//

// %%%% Algorthim %%%%

//

// preprocessor directives

// start main

// init variables

// prompt user to enter a grade from the set { A, B, C, D, F }:

// read in grade

// print you expect to get an (insert enter char) in this class

// end main

//

#include <stdio.h>

int main(void)

{

char grade = 0;

printf("Enter in a grade from the set { A, B, C, D, F }:");

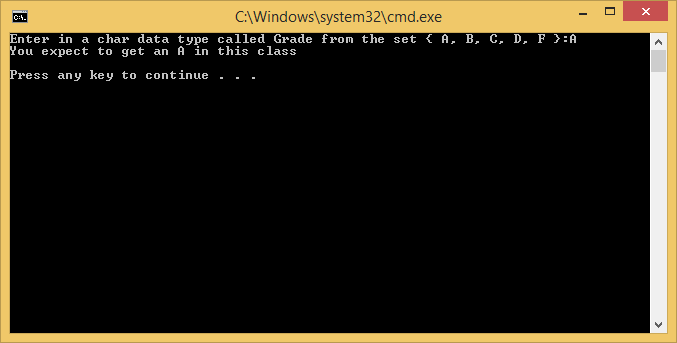
scanf\_s("%c", &grade);

printf("You expect to get an %c in this class", grade);

printf("\n\n");

return 0;

}



//

// HWK2(5).cpp

//

// By: Barak Barclay

// Date: 15 Sept 2015

//

// Problem Statement: Modify problem 4. Write a program using if statements. Write 5 if statements, one for each possible Grade entered. If true, the if statement will execute a puts statement that writes your letter grade to the monitor. Assume the user will enter a valid Grade.

//

//

// %%%% Algorthim %%%%

//

// preprocessor directives

// start main

// init variables

// prompt user to enter a grade from the set { A, B, C, D, F }:

// read in grade

// if grade = A

// puts You expect to get an A in this class

// end if

// if grade = B

// puts You expect to get an B in this class

// end if

// if grade = C

// puts You expect to get an C in this class

// end if

// if grade = D

// puts You expect to get an D in this class

// end if

// if grade = F

// puts You expect to get an F in this class

// end if

// end main

//

#include <stdio.h>

int main(void)

{

char grade = 0;

printf("Enter in a grade from the set { A, B, C, D, F }:");

scanf\_s("%c", &grade);

if (grade == 'A')

{

puts("\nYou expect to get an A in this class");

}

if (grade == 'B')

{

puts("\nYou expect to get an B in this class");

}

if (grade == 'C')

{

puts("\nYou expect to get an C in this class");

}

if (grade == 'D')

{

puts("\nYou expect to get an D in this class");

}

if (grade == 'F')

{

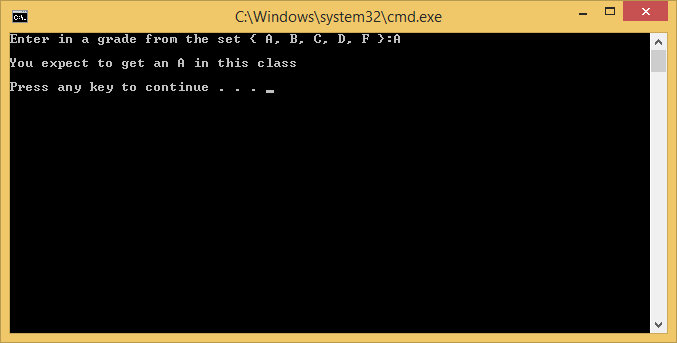
puts("\nYou expect to get an F in this class");

}

printf("\n");

return 0;

}



//

// HWK2(6).cpp

//

// By: Barak Barclay

// Date: 15 Sept 2015

//

// Problem Statement: Write a program to evaluate the logical equations given below. Find the answer by hand first, check by running the program.

// Given: int x = 0;

// int y = 33;

// bool b = false;

// bool c = true;

// bool d = true;

// bool e = -99;

//

// y = x && c && !b && d && (d || x) && e;

//

// y = (!b && (!x || e)) && (b || e || y);

//

//

//

// %%%% Algorthim %%%%

//

// preprocessor directives

// start main

// init variables

// do first equation

// print answer to first equation

// do second equation

// print answer to second equation

// end main

//

#include <stdio.h>

int main(void)

{

int x = 0;

int y = 33;

bool b = false;

bool c = true;

bool d = true;

bool e = -99;

y = x && c && !b && d && (d || x) && e;

printf("y = %d", y);

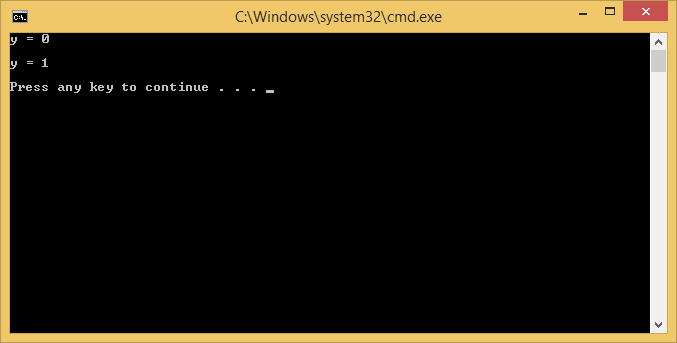
y = (!b && (!x || e)) && (b || e || y);

printf("\n\ny = %d", y);

printf("\n\n");

return 0;

}



//

// HWK2(7).cpp

//

// By: Barak Barclay

// Date: 15 Sept 2015

//

// Problem Statement: Write a program to evaluate the equations below. Find the answer by hand first, check by running the program.

// Given: int x1 = 0;

// int x2 = 2;

// int x3 = -7;

// int x4 = 11;

// int x5 = 3;

// int x6 = -8;

// int ans1 = 0;

// int ans2;

//

// ans1 = x1 \* (x2 \* x4) + x6 \* x5 – x3 \* x5 + (x3 + x6);

//

// ans2 = x1 + (x2 - x4) + x6 / x5 + x3 / x5 + (x6 + x3);

//

//

//

//

// %%%% Algorthim %%%%

//

// preprocessor directives

// start main

// init variables

// do first equation

// print answer to first equation

// do second equation

// print answer to second equation

// end main

//

#include <stdio.h>

int main(void)

{

int x1 = 0;

int x2 = 2;

int x3 = -7;

int x4 = 11;

int x5 = 3;

int x6 = -8;

int ans1 = 0;

int ans2;

ans1 = x1 \* (x2 \* x4) + x6 \* x5 - x3 \* x5 + (x3 + x6);

printf("ans1 = %d", ans1);

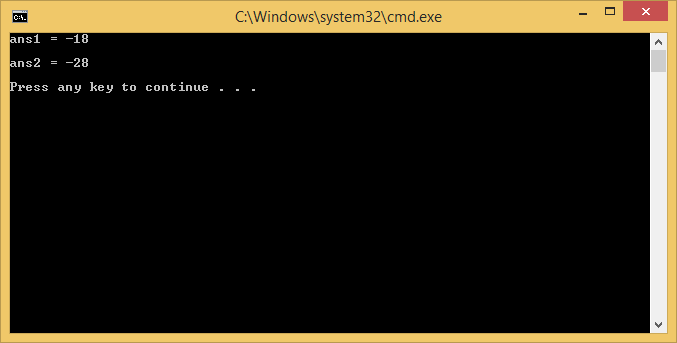
ans2 = x1 + (x2 - x4) + x6 / x5 + x3 / x5 + (x6 + x3);

printf("\n\nans2 = %d", ans2);

printf("\n\n");

return 0;

}



//

// HWK2(8).cpp

//

// By: Barak Barclay

// Date: 15 Sept 2015

//

// Problem Statement: Write a program to print out all the even numbers from 0 to 30 using a looping structure. Print the numbers across the screen (horizontally) with 2 spaces separating each number.

//

//

//

//

// %%%% Algorthim %%%%

//

// preprocessor directives

// start main

// init variables

// while loop

// print count and 2 spaces

// count++

// count++

// end while

// end main

//

#include <stdio.h>

int main(void)

{

int number = 30;

int count = 0;

while (count <= number)

{

printf("%d ", count);

count++;

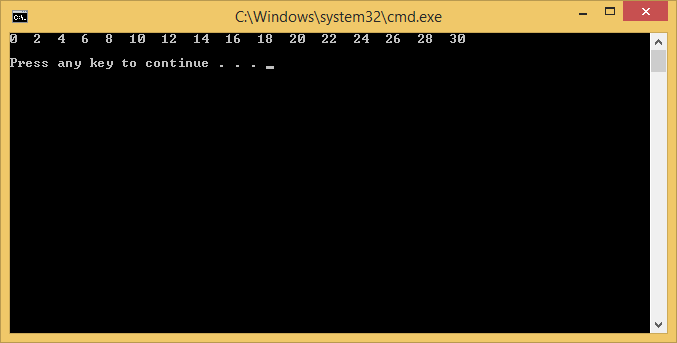
count++;

}

printf("\n\n");

return 0;

}



//

// HWK2(9).cpp

//

// By: Barak Barclay

// Date: 15 Sept 2015

//

// Problem Statement: Write a program that uses looping to print the following table of values. Use the tab escape sequence, \t, in the printf statement to separate the columns with tabs.

//

// N 10\*N 100\*N 1000\*N

//

// 1 10 100 1000

// 2 20 200 2000

// 3 30 300 3000

// 4 40 400 4000

// 5 50 500 5000

// 6 60 600 6000

// 7 70 700 7000

// 8 80 800 8000

// 9 90 900 9000

// 10 100 1000 10000

//

//

//

// %%%% Algorthim %%%%

//

// preprocessor directives

// start main

// init variables

// print N 10\*N 100\*N 100\*N

// while loop

// set variables eqaul to count times respected value

// print values

// count++

// end while

// end main

//

#include <stdio.h>

int main(void)

{

int number = 10;

int count = 1;

int N = 0;

int tentimesN = 0;

int hundredtimesN = 0;

int thousandtimeN = 0;

printf("N\t10\*N\t100\*N\t100\*N\n\n");

while (count <= number)

{

N = count;

tentimesN = count \* 10;

hundredtimesN = count \* 100;

thousandtimeN = count \* 1000;

printf("%d\t%d\t%d\t%d\n", N, tentimesN, hundredtimesN, thousandtimeN);

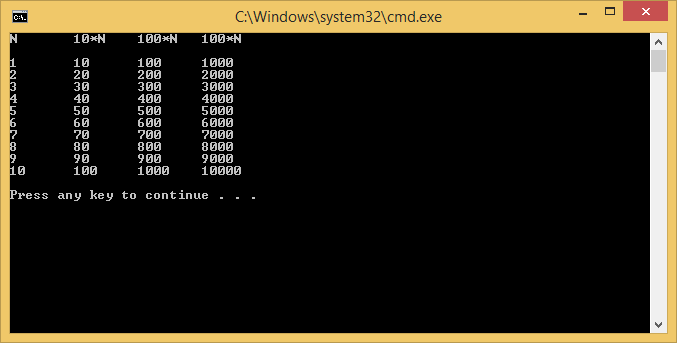
count++;

}

printf("\n");

return 0;

}



1. 4.1 a) definite. b) indefinite. c) control variable or counter. d) continue. e) break. f) switch selection statement