**Problem 1.1** *Compute division* (

**Graded manually Language: C**Fix the program below such that it prints the correct result. Why is the result 0.000? Write your answer and explanations within the comments.

#include <stdio.h>

int main() {  
double result; / ∗ The result of our calculation result = (3 + 1) / 5;  
printf("The value of 4/5 is %lf\n", result); return 0;

}

**Answer-**

#include <stdio.h>

int main() {

double result; /\*The result of our calculation\*/

result = (3.0 + 1.0) / 5;

printf("The value of 4/5 is %lf\n", result);

return 0;

}

//we put .0 because we are using double-operating

**Problem 1.2** *Wrong output*

**Language: C**

Fix the program below such that it prints the correct value. Why does the program print “The result is -1073745604”? (Values will vary). Write your answer and explanations within comments.

#include <stdio.h>

int main() {  
int result; / ∗ The result of our result=(2+7)∗ 9/3; printf("The result is %d\n"); return 0;

}

Answer-

#include <stdio.h>

int main() {

int result; /\*The result of our calculation\*/

result = (2 + 7) \* 9 / 3;

printf("The result is %d\n", result);

return 0;

}

//we have to add the result after the printf %d/n so

//the code understands you referring to the int result code

**Problem 1.3** *A compile error***Due by Monday, September 11th, 23:00**

**Language: C**

calculation

∗ /

You will get compiler errors when you try to compile the example code given below.  
Read the error messages that the compiler produces and fix the errors such that your source code compiles successfully. Then fix the program to print the correct result. Explain within comments the reason for the errors and describe your fixes.

include <stdio.h>

int main() {  
float result; // The result of the division

int a = 5;  
int b = 13.5;  
result = a / b;  
printf("The result is %d\n", result); return 0;

}

Answer-

#include <stdio.h>

int main() {

double result; /\* The result of the division \*/

int a = 5;

double b = 13.5;

result = a / b;

printf("The result is %lf\n", result);

return 0;

}

// i changed int b to double bc there decimal point

// and the result is from %d\n to Lf bc double uses this specifier

**Problem 1.4** *Simple arithmetics* **Language: C**

Write a program which does the following:

Assigns 17 to x and 4to y,

prints the values of x and y,

computes the sum of x and y and prints the result,

computes the product of x and y and prints the result,

computes the difference of x and y (x minus y) and prints the result,

computes the division of x and y (x divided by y) and prints the result (the result should be a float),

computes the remainder of the division of x and y in this order and prints the result.

**Testcase 1.4: Output**

x=17

y=4

sum=21

product=68

difference=13

division=4.250000

remainder of division=1

answer-

#include <stdio.h>

int main() {

int x = 17, y = 4;

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

printf("sum=%d\n", x+y);

printf("product=%d\n", x\*y);

printf("difference=%d\n", x-y);

printf("division=%.2f\n", (float) x/y);

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

return 0;

}

//compute the product means to multiply the int

**Problem1.5** *Using printf for multiple data types and conversions* Write a program which:

declares and initializes an integer variable x with 2138, and prints the value of x over 9 places,

declares and initializes a float variable y with −52.358925, and prints the value of y over 11 places and with a floating point precision of 5,

declares and initializes a char variable z with ’G’, and prints the character on the screen,

declares and initializes a double variable u with 61.295339687, and prints the value of u with a floating point precision of 7.

**Testcase 1.5: output**

x= 2138

y= -52.35892

z=G

u=61.2953

Answer-

#include <stdio.h>

int main(){

int x=2138;

printf("x=%9d\n", x);

float y= -52.358925;

printf("y=%11.5f\n", y); //floating point precision of 5 is decimal point after 5 place

char z= 'G';

printf("z=%c\n", z);

double u = 61.295339687;

printf("u=%.7lf\n", u);

return 0;

}

**Problem 1.6** *Printing a char as character and as decimal value*

**Language: C**

Write a program which declares and initializes a char variable c with ’F’ and prints on the screen the third character (within the alphabet) after c as a character and as the corresponding ASCII code using only arithmetic operations.

#include <stdio.h>

int main(){

char c = 'F';

printf("ASCII code=%d\n", c+3);

printf("C value= %c\n", c+3);

return 0;

}

/\*Problem 2.1 Reading from the keyboard Presence assignment

Write a program which does the following:

Graded automatically with testcases only

1. reads two doubles from the keyboard,

2. prints the sum of the two doubles,

3. prints the difference of the two doubles (first minus second), 4. prints the square of the first double,

5. reads two integers from the keyboard,

6. computes the sum and product of the two integers,

7. prints the sum and product of the integers,

8. reads two chars from the keyboard,

9. computes the sum and product of the two chars,

10. prints the sum and product of the chars as decimal values and as chars.

\*/

#include <stdio.h>

int main() {

double a , b;

scanf("%lf %lf", &a, &b);

printf("sum of doubles=%lf\n", a + b);

printf("difference of doubles=%lf\n", a - b);

printf("square=%lf\n", a \* a);

int c , d;

printf("sum of integers=%d\n", c + d);

scanf("%d %d", &c, &d);

printf("product of integers\n", c \* d);

// Compute the sum and product of the two integers

int sum\_int = c + d; // sum of the product

int product\_int = c \* d; // compute the product

// print the sum and product of the integers

printf("sum of the integers=%d\n", sum\_int);

printf("product if the integers=%d\n", product\_int);

// read two chars from the keyboard

char char1 , char2;

getchar();/\*Clearing the input buffer

After scanf(), a newline character is left in the buffer.

getchar() here reads and removes it.\*/

scanf("%c", &char1);

getchar();

scanf("%c", &char2);

// Compute the sum and product of the two chars

int sum\_char = char1 + char2;

int product\_char = char1 \* char2;

printf("sum of chars=%d\n", sum\_char);

printf("product of chars=%d\n", product\_char);

printf("sum of chars=%c\n", sum\_char);

printf("product of chars=%c\n", product\_char);

return 0;

}

/\*2.2 Write a program which does the following:

1. reads a char from the keyboard,

2. and prints the char as character as well as in decimal, octal and hexadecimal notation.

\*/

#include <stdio.h>

int main() {

char inputChar;

scanf("%c", &inputChar);

// Printing the character in different notations

// %c prints the character

// %d prints the ASCII value in decimal

// %o prints the ASCII value in octal

// %x prints the ASCII value in hexadecimal

printf("character=%c\n", inputChar);

printf("decimal=%d\n", inputChar);

printf("octal=%o\n", inputChar);

printf("hexadecmial=%x\n", inputChar);

return 0;

}

/\*Write a program where you can enter integer numbers for weeks, days and hours as input from the keyboard.

Your program should compute and output by printing on the screen the total number of hours.\*/

#include <stdio.h>

int main() {

int weeks, days, hours;

printf("enter the number of the weeks:");

scanf("%d", &weeks);

printf("enter the number of days:");

scanf("%d", &days);

printf("enter the number of hours");

scanf("%d", &hours);

//We should calculate the total number of hours

//The total hours are calculated by converting weeks to hours

//(1 week = 7 days, 1 day = 24 hours),

//adding the converted days to hours, and then adding the standalone hours.

int totalHours = (weeks \* 7 \* 24) + (days \* 24) + hours;

// Printing the total number of hours

printf("total number of hours:%d\n", totalHours);

return 0;

}

/\*Write a program that reads from the keyboard three

float values for the variables a, b and h.

Compute and print on the screen the areas of:

the square with the side a, the rectangle with

the length a and the width b, a triangle with the base a and the height h,

and a trapezoid with the bases a, b and the height h.\*/

#include <stdio.h>

int main() {

float a, b, h;

// Reading the float values for a, b, and h

scanf("%f %f %f", &a, &b, &h);

// Calculating the area of the square (side = a)

float areaSquare = a \* a;

// Calculating the area of the rectangle (length = a, width = b)

float areaRectangle = a \* b;

// Calculating the area of the triangle (base = a, height = h)

float areaTriangle = 0.5 \* a \* h;

// Calculating the area of the trapezoid (bases = a and b, height = h)

float areaTrapezoid = 0.5 \* (a + b) \* h;

return 0;

}

/\*Problem 2.5

Language: C

Write a program which reads an integer variable a from

the keyboard and prints the value on the screen.

Then declare and initialize a pointer ptr\_a pointing to a,

print the address contained in the pointer variable on the screen,

increase the value of a by 5 by using the pointer variable

and print the modified value and the address of

the variable on the screen as well.

You can safely assume that the input will be correct.

\*/

#include <stdio.h>

int main() {

int a; // Declare an integer variable 'a'

// Read an integer from the keyboard

printf("Enter an integer: ");

scanf("%d", &a);

// Print the value of 'a'

printf("The value of a: %d\n", a);

// declare and Initialize the pointer to point to 'a'

int \*ptr\_a = &a;

// Print the address contained in the pointer variable

printf("The address of a: %p\n", (void \*)ptr\_a);

// Increase the value of 'a' by 5 using the pointer variable

\*ptr\_a += 5;

// Print the modified value of 'a'

printf("The modified value of a: %d\n", a);

// Print the address of the variable 'a' again

printf("The address of a after modification: %p\n", (void \*)&a);

return 0;

}

/\*Problem 2.6 Multiple pointers to same data (1 point)

Due by Monday, September 18th, 23:00 Graded manually

Language: C

Write a program which reads two double variables x and y from the keyboard.

Then declare and initilize three pointers ptr\_one, ptr\_two

and ptr\_three such that ptr\_one and ptr\_two will both point

to the variable x and ptr\_three will point to y.

By using printf show that ptr\_one and ptr\_two contain the same memory address

and ptr\_three contains a different address.

You can assume that the input will be correct\*/

#include <stdio.h>

int main() {

double x , y;

scanf("%lf %lf", &x, &y);

// declare and Initialize the pointer to point to x

double \*ptr\_one , \*ptr\_two = &x;

// / declare and Initialize the pointer to point to y

double \*ptr\_three = &y;

// Print the memory addresses altogether double value x and y

printf("address of x=%p\n", (void \*)&x);

printf("address of y=%p\n", (void \*)&y);

// Print the memory addresses of each pointers

printf("address of ptr\_one=%p\n", (void \*)ptr\_one);

printf("address of ptr\_two=%p\n", (void \*)ptr\_two);

printf("address of ptr\_three=%p\n", (void \*)ptr\_three);

return 0;

}

/\*Problem 2.7 Infinite loop by bad coding

Language: C

The program below prints

i is 8 i is 8 ...

(1 point)

Graded manually

until you stop the execution by pressing Ctrl-C. Fix the program s

uch that it prints 8, 7, 6, 5 and 4 as values for i.

#include <stdio.h>

int main()

{

int i = 8;

while (i >= 4)

printf("i is %d\n", i);

i--;

printf("That’s it.\n");

return 0;

}

\*/

#include <stdio.h>

int main() {

int i = 8;

while (i >= 4) { // this starts the loop

printf("i is %d\n", i);

i--; // This needs to be inside the loop

}

printf("That's it.\n");

return 0;

}

/\*Problem 2.8 Divisible by 2 and 7?

Language: C

Write a program, where you can enter an integer from the keyboard.

Determine whether the number is divisible by both 2 and 7.

Then either print on the screen

“The number is divisible by 2 and 7” or

“The number is NOT divisible by 2 and 7”.

You can safely assume that the input will be valid.

Your solution has to satisfy the requirements from the problem description and has to pass the following testcase and potentially other testcases which are uploaded. All characters are relevant for passing testcases including newlines and spaces.

Testcase 2.8: input

56

Testcase 2.8: output

The number is divisible by 2 and 7

\*/

#include <stdio.h>

int main() {

int numbers;

// enter any integer

scanf("%d", &numbers);

// numbers that are only divisible by 2 and 7

if (numbers % 2 == 0 && numbers % 7 == 0) {

printf("the number is divisible by 2 and 7\n");

} else {

printf("the number is not divisible by 2 and 7\n");

}

return 0;

}

/\*

Problem 2.9 Categorization of characters

Language: C

Graded automatically with testcases only

Write a program where you can enter a character from the keyboard.

Then determine whether the character is a digit or a letter or

some other symbol and print a corresponding message on the screen.

You can safely assume that the input will be valid.

Your solution has to satisfy the requirements

from the problem description and has to pass the following

testcase and potentially other testcases which are uploaded.

All characters are relevant

for passing testcases including newlines and spaces.

Testcase 2.9: input

!

Testcase 2.9: output

! is some other symbol

\*/

#include <stdio.h>

#include <ctype.h> // Needed for isdigit() and isalpha()

int main() {

char any;

scanf("%c", &any);

if(isdigit(any)) { // isdigit is for digit or number

printf("%c is a digit\n", any);

} else if (isalpha(any)) { //isalpha is for character

printf("%c is a letter\n", any);

} else {

printf("%c is some other symbol\n", any);

}

return 0;

}

/\*Problem 2.10 Days and hours

Language: C

Write a program where you can enter

an integer n from the keyboard. T

hen a conversion table for 1 to n days

should be printed on the screen as

in the example below.

Make sure that the integer value

you entered for n is valid

(positive and non-zero).

If an invalid integer n was

entered then repeat the entering

until a valid value will be entered.

Use a while loop in your solution.

1 day = 24 hours

2 days = 48 hours

3 days = 72 hour

\*/

#include <stdio.h>

int main() {

int n;

while(1) { // The program then enters a while loop that will keep looping

// until a valid input is received

printf("enter a positive non-zero integer(n):");

scanf("%d", &n);

if (n > 0) { // The program then checks if the input is a positive integer

break; // exist the loop if n is valid

} else { /\* If the input is not valid, the program prints an error message

and asks the user to try again\*/

printf("invalid input. try again to enter a positive non-zero integer\n");

}

}

// print the conversion table

printf("conversion table:\n");

for (int i = 1; i <= n; i++) {

printf("%d day%s = %d hours%s\n", i, (i == 1) ? "" : "s", i \* 24, (i == 1) ? "" : "s");

}

return 0;

}

/\*

Problem 3.1 Writing numbers (1 point) Presence assignment, Language: C

Write a program where you first enter a float x and then an integer n from the keyboard. Print the float x n times to the screen. Make sure that n will have a valid integer value. In the invalid case repeat entering n until a valid value will be entered.

Your solution has to satisfy the requirements from the problem description and has to pass the following testcase and potentially other testcases which are uploaded. All characters are relevant for passing testcases including newlines and spaces.

Testcase 3.1: input

1.25

-8

0

4

Testcase 3.1: output

Input is invalid, reenter value

Input is invalid, reenter value

1.250000

1.250000

1.250000

1.250000

\*/

#include <stdio.h>

int main() {

float x;

int n;

// Enter the float x

scanf("%f", &x);

// Enter the integer n, with validation

do {

scanf("%d", &n);

if (n <= 0) {

printf("Input is invalid, reenter value\n");

}

} while (n <= 0);

// Print the float x n times

for (int i = 0; i < n; i++) {

printf("%.6f\n", x);

}

return 0;

}

/\*Problem 3.2

Language: C

Writing characters

Write a program where you first enter a lowercase character ch

and then an integer n from the keyboard.

Print the characters ch, ch−1, ..., ch−n on the screen.

You can safely assume that the input is valid and you do not have to do any checks.

\*/

#include <stdio.h>

int main() {

char ch;

int n;

scanf("%c", &ch);

scanf("%d", &n);

printf("here are the characters:\n");

for (int i = 1; i <= n; i++) {

printf("%c\n", ch - i); // i added a loop that counts from 0 to n

}

printf("\n");

return 0;

}

/\*Problem 3.3 Centimeters to kilometers

Language: C

Write a program that converts an integer length that is entered

from the keyboard in cm to km. Write and use afunction float convert(int cm)

that does the actual conversion.

You can safely assume that the input will be valid.

Your solution has to satisfy the requirements from the problem description and has to pass the following testcase and potentially other testcases which are uploaded. All characters are relevant for passing testcases including newlines and spaces.

Testcase 3.3: input

12

Testcase 3.3: output

Result of conversion: 0.000120

\*/

#include <stdio.h>

float convert(int cm) {

return (float) cm/10000.0; // Convert cm to km

}

int main() {

int lengthInCm;

scanf("%d", &lengthInCm); // Read the length in centimeters

float lengthInKm = convert(lengthInCm); // Convert to kilometers

printf("result of conversion: %.6f \n", lengthInKm); // Print the result with 6 decimal places

return 0;

}

/\* Problem 3.4 Wrong position

Language: C

The program below should print the position of the f

irst occurrence of the character ’g’ within a string.

You can safely assume that ’g’ will be contained in the string.

Why does it always print the position 0? Fix the program such that it prints the correct position.

\*/

#include <stdio.h>

// Updated position function to manually search for the character in the string

int position(char str[], char c) {

int idx;

for (idx = 0; str[idx] != '\0'; ++idx) {

if (str[idx] == c) {

return idx;

}

}

return -1;

}

int main() {

char line[80];

while (1) {

printf("enter string:\n");

fgets(line, sizeof(line), stdin);

// Replace '\n' with '\0' to avoid incorrect results when searching for the character

for (int i = 0; line[i] != '\0'; i++) {

if (line[i] == '\n') {

line[i] = '\0';

break;

}

}

int result = position(line, 'g');

if (result != -1) {

printf("the first occurrence of 'g' is at position: %d\n", result);

} else {

printf("'g' not found in the string.\n");

}

}

return 0;

}

/\*Write a c program where you first enter a character c

followed by an integer n and n double values representing temperatures in Celsius.

Use an array for storing the temperatures. You can assume that not more than 100 temperature values would be entered.

Your program should compute and/or print the following: if c is ’s’ the sum of the temperatures,

if c is ’p’ the list of all tem- peratures, if c it ’t’ the list

of all temperatures in Fahrenheit and if another character

was entered then the arithmetic mean (or average) of all temperatures.

The formula for converting Celsius to Fahrenheit is the following:

F = 5/9 · C + 32. Use a switch instruction in your solution.

You can safely assume that the input will be valid.\*/

#include <stdio.h>

int main() {

char c;

int n;

double temperatures[100]; // Array to hold up to 100 temperatures

printf("Enter a character and an integer: ");

scanf(" %c %d", &c, &n); // Note the space before %c to catch any previous newline

printf("Enter %d temperatures in Celsius: ", n);

for (int i = 0; i < n; i++) {

scanf("%lf", &temperatures[i]);

}

switch (c) {

case 's': { // Sum of temperatures

double sum = 0;

for (int i = 0; i < n; i++) {

sum += temperatures[i];

}

printf("Sum of temperatures: %f\n", sum);

break;

}

case 'p': { // Print all temperatures

printf("List of temperatures: ");

for (int i = 0; i < n; i++) {

printf("%f ", temperatures[i]);

}

printf("\n");

break;

}

case 't': { // Convert to Fahrenheit and print

printf("Temperatures in Fahrenheit: ");

for (int i = 0; i < n; i++) {

double fahrenheit = temperatures[i] \* 9.0 / 5.0 + 32;

printf("%f ", fahrenheit);

}

printf("\n");

break;

}

default: { // Arithmetic mean

double sum = 0;

for (int i = 0; i < n; i++) {

sum += temperatures[i];

}

double mean = sum / n;

printf("Arithmetic mean of temperatures: %f\n", mean);

break;

}

}

return 0;

}

/\*Problem 3.6 Kilograms and grams to pounds

Language: C

Write a program that converts the units of mass

kg and g to pounds.

First read the weight of an object expressed

by values for kilograms and grams

from the keyboard and convert the units of mass using the function (written by you as well)

float to\_pounds(int kg, int g);

that does the actual conversion. Note that:

1 kilogram = 2.2 pounds

You can safely assume that the input will be valid.

Your solution has to satisfy the requirements from the problem description

and has to pass the following testcase and potentially other testcases which are uploaded.

All characters are relevant for passing testcases including newlines and spaces.

Testcase 3.6: input

5

100

Testcase 3.6: output

Result of conversion: 11.220000

\*/

#include <stdio.h>

float to\_pounds(int kg, int g) {

//convert total weight to kilograms (including the grams part)

float totalKg = kg + g / 1000.0f;

return totalKg \* 2.2f; //convert to pound

}

int main() {

int kg, g;

scanf("%d %d", &kg, &g);

printf("enter the weight in kilograms and grams:\n");

float pounds = to\_pounds(kg, g);

printf("result of conversion: %.6f\n", pounds);

return 0;

}

/\*Problem 3.7 Printing a form

Language: C

Write a program which reads two integers n, m and a character c from the keyboard.

This pro- gram should define and call a function with the prototype:

void print form(int n, int m, char c);

which prints a trapezoid of height n, bases m and m+n-1 consisting of the character c as in the following testcase.

Testcase 3.7: input

4

3

$

Testcase 3.7: output

$$$

$$$$

$$$$$

$$$$$$ \*/

#include <stdio.h>

void print\_form(int n, int m, char c) {

for (int i = 0; i < n; i++) {

for (int j = 0; j < m + i; j++) {

printf("%c", c);

}

printf("\n");

}

}

int main() {

int n;

int m;

char c;

scanf("%d %d %c", &n, &m, &c);

print\_form(n, m, c);

return 0;

}

/\*Problem 3.8 Computing sum and average

Language: C

Write a program where you can enter from the keyboard up to 10 floats.

If the number entered is equal to −99.0, stop reading numbers from the keyboard and compute the sum

and average of all values (excluding −99.0) using two functions (one for the sum and one for the average).

Print your results on the screen.

You can safely assume that the input will be valid.

Make sure you consider all the cases: less than 10 numbers might be entered.

After all the numbers have been entered you need to make sure that the sum and average are computed.

\*/

#include <stdio.h>

#define MAX\_NUMS 10

// Function to compute the sum of float array

float compute\_sum(float arr[], int length) {

float sum = 0.0;

for (int i = 0; i < length; i++) {

sum += arr[i];

}

return sum;

}

// Function to compute the average of float array

float compute\_average(float arr[], int length) {

if (length == 0) {

return 0; // To avoid division by zero

}

return compute\_sum(arr, length) / length;

}

int main() {

float numbers[MAX\_NUMS];

float number;

int count = 0;

printf("Enter up to 10 float numbers (enter -99.0 to stop):\n");

while (count < MAX\_NUMS) {

scanf("%f", &number);

if (number == -99.0) {

break;

}

numbers[count++] = number;

}

float sum = compute\_sum(numbers, count);

float average = compute\_average(numbers, count);

printf("Sum: %f\n", sum);

printf("Average: %f\n", average);

return 0;

}

/\*Problem 3.9 Determine sum of two values in array

Language: C

Write a program which reads from the keyboard an integer value n followed by n double values as elements of an array with not more than 20 elements.

Write also a function with the prototype: double sum25(double v[], int n);

which computes and returns the sum of the elements in v at position 2 and position 5.

Check that positions 2 and 5 are valid within v, if not then print a corresponding message on the screen.

In this case the function should return the value −111. Print the sum on the screen.

Your solution has to satisfy the requirements from the problem description

and has to pass the following testcase and potentially other testcases which are uploaded.

All characters are relevant for passing testcases including newlines and spaces.

Testcase 3.9: input

6

1.5

1.5

1.5

1.5

1.5

-1.5

Testcase 3.9: output

sum=0.000000

\*/

#include <stdio.h>

// Function to calculate the sum of the 3rd and 6th elements in the array

double sum25(double v[], int n) {

// Check if the array has at least 6 elements

if (n < 5) {

printf("One or both positions are invalid\n");

return -111.0; // Return a special error value if not

}

// Calculate the sum of the 3rd (index 2) and 6th (index 5) elements

double sum = v[2] + v[5];

return sum; // Return the sum

}

int main() {

int n; // Variable to store the number of elements to read

scanf("%d", &n); // Read the number of elements

// Check if there are at least 6 elements

if (n < 6) {

printf("One or both positions are invalid\n");

return 1; // Exit the program if not

}

double v[20]; // Array to store the input values

// Read `n` values into the array

for (int i = 0; i < n; i++) {

scanf("%lf", &v[i]);

}

// Calculate the sum of the 3rd and 6th elements

double result = sum25(v, n);

// Check if the result is not the error value

if (result != -111.0) {

printf("sum=%lf\n", result); // Print the sum if valid

}

return 0;

}

/\*Problem 3.10 Return changes by reference

Language: C

Write a program which reads two float values from the keyboard.

Then write three functions.

The first function should return the product of the two float values and should have the proto-type:

float product(float a, float b);

The second function should return the product of the two float values by reference and should have the prototype:

void productbyref(float a, float b, float \*p);

The third function should add 3 to the first float and 11 to the second float and return the change by reference.

It should have the prototype:

void modifybyref(float \*a, float \*b);

Show that the calls of the first two functions have the same effect. Also show what is the effect of calling modifybyref.

You can safely assume that the input will be valid.

\*/

#include <stdio.h>

// Function to return the product of two floats

float product(float a, float b) {

return a \* b; // Multiply a and b and return the result

}

// Function to calculate the product of two floats and store it in a reference (pointer)

void productbyref(float a, float b, float \*p) {

\*p = a \* b; // Multiply a and b and store the result in the location pointed to by p

}

// Function to modify two float values by reference

void modifybyref(float \*a, float \*b) {

\*a += 3.0; // Add 3.0 to the value pointed by a

\*b += 11.0; // Add 11.0 to the value pointed by b

}

int main() {

float num1, num2;

// Read two float values from the user

scanf("%f %f", &num1, &num2);

float result1, result2;

// Calculate the product of num1 and num2 using the product function

result1 = product(num1, num2);

// Calculate the product of num1 and num2 using the productbyref function

productbyref(num1, num2, &result2);

// Print the results of the product and productbyref functions

printf("result of product function: %.6f\n", result1);

printf("result of productbyref function: %.6f\n", result2);

// Modify num1 and num2 by calling modifybyref

modifybyref(&num1, &num2);

// Print the modified values of num1 and num2

printf("modified value of the first float: %.6f\n", num1);

printf("modified value of the second float: %.6f\n", num2);

return 0;

}

/\*Problem 3.11 Working with strings (2 points) Due by Monday, September 25th, 23:00 Graded automatically with testcases only

Language: C

Write a program where you can enter two strings called one and two from the keyboard. The string should be able to contain spaces. The program should do the following:

1. Print on the screen the lengths of both strings,

2. Print on the screen the concatenation of one with two,

3. Declare a third string, copy correctly two into it and print the third string to the screen,

4. Compare the two strings two and one and print a corresponding message to the screen,

5. Read a character c from the keyboard and search for c in two. The position of the first occurrence of c within two should be printed to the screen. If the character is not contained in the string then print a corresponding message on the screen.

For solving this problem use the string functions from string.h.

Learn how to use them with the help of the man pages.

Your solution has to satisfy the requirements from the problem description and has to pass the following testcase and potentially other testcases which are uploaded. All characters are relevant for passing testcases including newlines and spaces.

Testcase 3.11: input

first string

hello world

l

Testcase 3.11: output

length1=12

length2=11

concatenation=first stringhello world

copy=hello world

one is smaller than two

position=2\*/

#include <stdio.h>

#include <string.h>

int main() {

// Initialize character arrays to hold two input strings and one result string

char one[100];

char two[100];

char result[100];

// Read the first string from the user

printf("enter the first string: ");

fgets(one, sizeof(one), stdin);

one[strcspn(one, "\n")] = '\0'; // Remove the newline character at the end

// Read the second string from the user

printf("enter the second string: ");

fgets(two, sizeof(two), stdin); // to read two strings from the user, removing the newline character from each input

two[strcspn(two, "\n")] = '\0'; // Remove the newline character at the end

// Print the lengths of the first and second strings

printf("length of the first string: %zu\n", strlen(one));

printf("length of the second string: %zu\n", strlen(two));

// Concatenate the first and second strings into 'result'

strcpy(result, one);

strcat(result, two);

printf("concatenation of the first and second strings: %s\n", result);

// Copy the second string into 'result'

strcpy(result, two);

printf("copy of the second string: %s\n", result);

// Compare the two strings and print if they are equal or not

int compareResult = strcmp(two, one);

if (compareResult == 0) {

printf("the two strings are equal.\n");

} else {

printf("the two strings are not equal.\n");

}

// Read a character from the user and search for it in the second string

char c;

printf("enter a character to search in the second string: ");

scanf(" %c", &c);

char \*position = strchr(two, c);

if (position != NULL) {

int index = position - two;

printf("character '%c' found at position %d in the second string.\n", c, index);

} else {

printf("character '%c' not found in the second string.\n", c);

}

return 0;

}

/\*Problem P.1 (2 points)

Language: C

Please complete the following program fragment

such that it prints n rows with the pattern below using nested loops.

int n;

scanf("%d", &n);

// add your code completion below

So if the user enters 6 for n, the following 6 rows should be printed:

A

AB

ABC

ABCD

ABCDE

ABCDEF

\*/

#include <stdio.h>

int main() {

int n;

scanf("%d", &n);

// Outer loop for each row

for (int i = 0; i < n; i++) {

// Inner loop for printing characters in each row

for (int j = 0; j <= i; j++) {

printf("%c", 'A' + j); // Print character starting from 'A'

}

printf("\n"); // New line after each row

}

return 0;

}

/\*Problem P.2

Language: C

Write a program which does the following:

a) reads a double from the keyboard,

b) reads a float from the keyboard,

c) reads an integer from the keyboard,

d) stores the product of these three values into the variable result,

No information should be lost.

e) prints the value of result,

f) uses a pointer r ptr to add 5 to result,

g) prints the new values twice, once by using result, once by using r ptr.

\*/

#include <stdio.h>

int main() {

double a;

printf("Enter a double value:");

scanf("%f", &a);

float b;

printf("enter a float value:");

scanf("%lf", &b);

int x;

printf("enter a integer:");

scanf("%d", &x);

double result; // used double bec, it holds more digits

double \*r\_ptr;

result = a \* b \* x;

r\_ptr = &result;

\*r\_ptr += 5;

printf("Value of result (after adding 5): %lf\n", result);

printf("Value of result (after adding 5) using r\_ptr: %lf\n", \*r\_ptr);

return 0;

}