Qno.1)

#include <stdio.h>

int \*biggest\_of\_two(int \*a, int \*b) {

return \*a > \*b ? a : b;

}

int \*biggest\_of\_three(int \*a, int \*b, int \*c) {

int \*max\_ab = biggest\_of\_two(a, b);

return \*max\_ab > \*c ? max\_ab : c;

}

int main() {

int a, b, c;

int \*biggest;

printf("Enter three integers separated by space: ");

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

biggest = biggest\_of\_two(&a, &b);

printf("Biggest of two: %d\n", \*biggest);

biggest = biggest\_of\_three(&a, &b, &c);

printf("Biggest of three: %d\n", \*biggest);

return 0;

}

Qno.2)

#include <stdio.h>

void div\_rem(int a, int b, int \*quotient, int \*remainder) {

\*quotient = a / b;

\*remainder = a % b;

}

int main() {

int a, b, quotient, remainder;

printf("Enter two integers separated by space: ");

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

div\_rem(a, b, &quotient, &remainder);

printf("Quotient: %d\n", quotient);

printf("Remainder: %d\n", remainder);

return 0;

}

Qno.3)

#include <stdio.h>

int string\_length(char \*str) {

int length = 0;

while (\*str != '\0') {

length++;

str++;

}

return length;

}

int main() {

char input[100];

printf("Input a string: ");

fgets(input, sizeof(input), stdin);

// Removing the newline character from the input

input[string\_length(input) - 1] = '\0';

int length = string\_length(input);

printf("The length of the given string \"%s\" is: %d\n", input, length);

return 0;

}

Qno.4)

#include <stdio.h>

#include <string.h>

void reverse\_string(char \*str) {

int length = strlen(str);

for (int i = length - 1; i >= 0; i--) {

printf("%c", \*(str + i));

}

}

int main() {

char input[100];

printf("Input a string: ");

fgets(input, sizeof(input), stdin);

// Removing the newline character from the input

input[strlen(input) - 1] = '\0';

printf("Reverse of the string is: ");

reverse\_string(input);

printf("\n");

return 0;

}

Qno.5)

#include <stdio.h>

#include <stdbool.h>

void count\_words\_and\_characters(char \*sentence, int \*word\_count, int \*char\_count) {

bool in\_word = false;

int word\_len = 0;

while (\*sentence != '\0') {

if (\*sentence == ' ' || \*sentence == '\n') {

if (in\_word) {

char\_count[\*word\_count] = word\_len;

(\*word\_count)++;

word\_len = 0;

in\_word = false;

}

} else {

word\_len++;

in\_word = true;

}

sentence++;

}

if (in\_word) {

char\_count[\*word\_count] = word\_len;

(\*word\_count)++;

}

}

int main() {

char input[100];

int word\_count = 0;

int char\_count[100];

printf("Enter a sentence: ");

fgets(input, sizeof(input), stdin);

count\_words\_and\_characters(input, &word\_count, char\_count);

printf("Total number of words: %d\n", word\_count);

printf("Number of characters for each: ");

for (int i = 0; i < word\_count; i++) {

printf("%d ", char\_count[i]);

}

printf("\n");

return 0;

}

Qno.6)

#include <stdio.h>

#include <string.h>

void process\_sentence(char \*input, char \*output) {

while (\*input != '\0') {

if (\*input != ' ' && \*input != '\n') {

if (\*input == '.') {

\*output = ':';

} else {

\*output = \*input;

}

output++;

}

input++;

}

\*output = '\0';

}

int main() {

char input[100];

char output[100];

printf("Enter a sentence: ");

fgets(input, sizeof(input), stdin);

process\_sentence(input, output);

printf("Processed sentence: %s\n", output);

return 0;

}

Qno.7)

#include <stdio.h>

#include <string.h>

void concatenate\_strings(char \*str1, char \*str2, char \*result) {

while (\*str1 != '\0') {

\*result = \*str1;

str1++;

result++;

}

while (\*str2 != '\0') {

\*result = \*str2;

str2++;

result++;

}

\*result = '\0';

}

int main() {

char str1[100], str2[100], result[200];

printf("Enter the first string: ");

fgets(str1, sizeof(str1), stdin);

str1[strlen(str1) - 1] = '\0'; // Remove the newline character

printf("Enter the second string: ");

fgets(str2, sizeof(str2), stdin);

str2[strlen(str2) - 1] = '\0'; // Remove the newline character

concatenate\_strings(str1, str2, result);

printf("Concatenated string: %s\n", result);

return 0;

}

Qno.8)

#include <stdio.h>

void factorial(int n, long long \*result) {

\*result = 1;

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

\*result \*= i;

}

}

int main() {

int number;

long long fact;

printf("Input a number: ");

scanf("%d", &number);

factorial(number, &fact);

printf("%d's factorial: %lld\n", number, fact);

return 0;

}

Qno.9)

#include <stdio.h>

void get\_multiple\_values(int \*values) {

values[0] = 1;

values[1] = 2;

values[2] = 3;

}

int main() {

int values[3];

get\_multiple\_values(values);

printf("Values: %d, %d, %d\n", values[0], values[1], values[2]);

return 0;

}

Qno.10)

#include <stdio.h>

#include <string.h>

#include <stdlib.h>

#define MAX\_NAME\_LENGTH 50

#define MAX\_NAMES 100

int compare\_names(const void \*a, const void \*b) {

return strcmp((const char \*)a, (const char \*)b);

}

int main() {

char names[MAX\_NAMES][MAX\_NAME\_LENGTH];

int n;

printf("Enter the number of names: ");

scanf("%d", &n);

getchar(); // To remove the newline character after reading the number

printf("Enter names:\n");

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

fgets(names[i], MAX\_NAME\_LENGTH, stdin);

names[i][strcspn(names[i], "\n")] = '\0'; // Remove the newline character

}

qsort(names, n, MAX\_NAME\_LENGTH, compare\_names);

printf("Alphabetical list:\n");

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

printf("%s\n", names[i]);

}

return 0;

}