**Lesson 8**

**Prectice exercises**

**1)**

**#include** <stdio.h>

**#include** <math.h>

**int** **main**(**void**) {

**int** n;

**scanf**("%i", &n);

**int** i = 0;

**printf**("%i = ", n);

**do**{

**int** oct = n%8;

n/=8;

**printf**("+ (%i \* 8^%i) ", oct, (**int**) i);

i++;

}**while**(n>0);

**printf**("\nDone");

**return** 0;

}

**2)**

**#include** <stdio.h>

**#include** <math.h>

**int** **main**(**void**) {

**int** n;

**scanf**("%i", &n);

**int** i = 0;

**printf**("%i = ", n);

**do**{

**int** hex = n%16;

n/=16;

**printf**("+ (%i \* 16^%i) ", hex, (**int**) i);

i++;

}**while**(n>0);

**printf**("\nDone");

**return** 0;

}

**3)**

**#include** <stdio.h>

**#include** <math.h>

**int** **main**(**void**) {

**int** n, b;

**scanf**("%i\t%i", &n, &b);

**int** i = 0;

**printf**("%i = ", n);

**do**{

**int** exp = n%b;

n/=b;

**printf**("+ (%i \* %i^%i) ", exp, b, (**int**) i);

i++;

}**while**(n>0);

**printf**("\nDone");

**return** 0;

}

**Lesson 7**

**Prectice exercises**

**4)**

**#include** <stdio.h>

**#include** <math.h>

**int** **main**(**void**) {

**int** i, j, z;

**for**(i = 0, z = 12; i < 8; i++, z++){

**for**(j = z;j > 0; j--){

**printf**("\*");

}

**printf**("\n");

}

**for**(i = 0; i < 5; i++){

**for**(j = z; j > 0; j--){

**printf**("\*");

}

**printf**("\n");

}

**for**(i = 0; i < 19; i++, z--){

**for**(j = z;j > 0; j--){

**printf**("\*");

}

**printf**("\n");

}

**for**(i = 0; i < 4; i++){

**printf**("\*\n");

}

**for**(i = 0, z = 2; i < 7; i++, z++){

**for**(j = z;j > 0; j--){

**printf**("\*");

}

**printf**("\n");

}

**return** 0;

}

**Lesson 10**

**Prectice exercises**

**1)**

**#include**<stdio.h>

**int** **main**(**void**){

**char** ch;

**int** count = 0;

**do**{

ch = **getchar**();

**putchar**(ch);

**putchar**('\n');

**if**(ch == 'a'){

count++;

}

}**while**(ch != '\n');

**printf**("%i", count);

**return** 0;

}

**3) and 4)**

**#include**<stdio.h>

**int** **main**(**void**){

**char** ch;

**int** count = 0;

**int** max = 0;

**printf**("Input some characters: ");

**do**{

ch = **getchar**();

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

count++;

}**else**{

**printf**("The length of the word is %i\n\n", count);

**if** (max < count){

max = count;

}

count = 0;

}

}**while**(ch != '\n');

**printf**("The longest word has a length of %i\n", max);

**printf**("Done");

**return** 0;

}

**Lesson 11**

**Writing file**

**#include**<stdio.h>

**int** **main**(**void**){

FILE \*file;

file = **fopen**("datafile", "w");

**if**(file == NULL){

**printf**("Problem opening file.");

**return** 1;

}

**printf**("File opened successfully.");

**fprintf**(file, "Hello there!");

**fclose**(file);

**return** 0;

}

**Prectice exercises**

**3) and 4)**

**#include**<stdio.h>

**int** **main**(**void**){

FILE \*val, \*story, \*news;

val = **fopen**("values.txt", "r");

story = **fopen**("story.txt", "r");

news = **fopen**("newStory.txt", "w");

**if**(val == NULL || story == NULL){

**printf**("Problem opening file.\n");

**return** 1;

}

**char** word[81];

**char** ch;

**do**{

ch = **getc**(story);

**if**(ch == '$'){

**fscanf**(val, "%s", word);

**printf**("%s", word);

**fprintf**(news, "%s", word);

}**else**{

**printf**("%c", ch);

**fprintf**(news, "%c", ch);

}

}**while**(!**feof**(val));

**printf**("File opened successfully.");

**fclose**(val);

**fclose**(story);

**fclose**(news);

**return** 0;

}

**Lesson 12**

**Prectice exercises**

**1)**

**#include**<stdio.h>

**int** **main**(**void**){

**int** arr[10];

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

**if**(i == 0){

arr[i] = 1;

}**else**{

arr[i] = arr[i-1] \* i;

}

}

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

**printf**("%i\t", arr[i]); }

**return** 0;

}

**2) and 3)**

**#include**<stdio.h>

**#include**<stdlib.h>

**int** **main**(**void**){

**int** arr[20];

**int** min = 100, max = 50, sum = 0, average = 0;

**for**(**int** i = 0; i<20; i++){

arr[i] = **rand**()%100;

**if**(arr[i] < 50){

i--;

**continue**;

}

**if**(arr[i] > max){

max = arr[i];

}

**if**(arr[i] < min){

min = arr[i];

}

sum += arr[i];

}

average = sum / 20;

**for**(**int** i = 0; i<20; i++){

**printf**("%i\t", arr[i]);

}

**printf**("\nThe minimum value is %i.\nThe maximum value is %i.\n", min, max);

**printf**("The average is %i", average);

**return** 0;

}

**Lesson 13**

**Prectice exercises**

**1)**

**#include**<stdio.h>

**#include**<stdlib.h>

**int** **main**(**void**){

**int** matrix[12][12];

**for**(**int** i = 1; i <= 12; i++){

**for**(**int** j = 1; j <= 12; j++){

matrix[i][j] = i \* j;

}

}

**for**(**int** i = 1; i <= 12; i++){

**for**(**int** j = 1; j <= 12; j++){

**printf**("%i\t", matrix[i][j]);

}

**printf**("\n");

}

**return** 0;

}

**2)**

**#include**<stdio.h>

**#include**<stdlib.h>

**int** **main**(**void**){

**int** R = 3, C = 4;

**int** a[R][C], b[C][R];

**int** i, j, val = 100;

/\* set matrix elements \*/

**for** (i = 0; i < R; i++) {

**for** (j = 0; j < C; j++) {

a[i][j] = val;

val++;

}

}

val = 100;

**for** (i = 0; i < C; i++) {

**for** (j = 0; j < R; j++) {

b[i][j] = a[j][i];

b[i][j] += 25;

val++;

}

}

/\* print matrix elements \*/

**for** (i = 0; i < R; i++) {

**for** (j = 0; j < C; j++) {

**printf**("%5d", a[i][j]);

}

**printf**("\n");

}

**printf**("\n");

**for** (i = 0; i < C; i++) {

**for** (j = 0; j < R; j++) {

**printf**("%5d", b[i][j]);

}

**printf**("\n");

}

**return** 0;

}

**3)**

**#include**<stdio.h>

**#include**<stdlib.h>

**int** **main**(**void**){

**int** R = 3, C = 4;

**int** a[R][C];

**int** i, j, val = 100, max = 0, r,c;

**for** (i = 0; i < R; i++) {

**for** (j = 0; j < C; j++) {

a[i][j] = **rand**()&100;

**if**(a[i][j] > max){

max = a[i][j];

r = i + 1;

c = j + 1;

}

val++;

}

}

**for** (i = 0; i < R; i++) {

**for** (j = 0; j < C; j++) {

**printf**("%5d", a[i][j]);

}

**printf**("\n");

}

**printf**("The maximum value is %i and position is on row %i and column %i.\n", max, r,c);

**return** 0;

}

**Lesson 14**

**#include**<stdio.h>

**#include**<stdlib.h>

**typedef** **struct**{

**int** customerID;

**\_Bool** isSavings;

**int** balance;

**double** interestRate;

}account;

**int** **main**(**void**){

account myAcct;

**int** time;

**printf**("Write your account data.\n");

**scanf**("%i%i%i", &myAcct.customerID, &myAcct.isSavings, &myAcct.balance);

**printf**("\nCustomer ID is: %i.\n", myAcct.customerID);

**if**(myAcct.isSavings){

**printf**("Account type is savings.\n");

myAcct.interestRate = 0.02;

}**else**{

**printf**("Account type is not savings.\n");

myAcct.interestRate = 0.01;

}

**printf**("Initial balance is: %i.\n", myAcct.balance);

**printf**("How long do you want to wait?\n");

**scanf**("%i", &time);

**for**(**int** i = 0; i < time; i++){

myAcct.balance = myAcct.balance\*(1 + myAcct.interestRate);

}

//myAcct.balance = myAcct.balance + myAcct.interestRate\*time;

**printf**("Resulting balance is: %i.\n", myAcct.balance);

**return** 0;

}

**Prectice exercises**

**2)**

**#include**<stdio.h>

**#include**<stdlib.h>

**typedef** **struct**{

**int** studentID;

**char** gender;

**double** labs;

**double** quizzes;

**double** assignment;

**double** exam;

**double** finalGrade;

}student;

**int** **main**(**void**){

student st;

**printf**("Enter a student ID: ");

**scanf**("%i", &st.studentID);

**printf**("\nEnter a student gender: ");

**scanf**(" %c", &st.gender);

**printf**("\nEnter results of each assessment in order: labs, quizzes, assignment and exam. ");

**scanf**("%lf%lf%lf%lf", &st.labs, &st.quizzes, &st.assignment, &st.exam);

st.finalGrade = 0.25\*st.labs + 0.35\*st.quizzes + 0.05\*st.assignment + 0.35\*st.exam;

**printf**("The final grade is %f", st.finalGrade);

**return** 0;

}

**Lesson 15**

**Prectice exercises**

**1)**

**#include**<stdio.h>

**#include**<stdlib.h>

**typedef** **struct**{

**int** r, g, b;

}Color;

**typedef** **struct**{

**int** x, y;

}Cords;

**typedef** **struct**{

Cords upperLeftCorner;

**int** sideLength;

Color outline;

Color fill;

}Square;

**int** **main**(**void**){

Color white = {255, 255, 255};

Color red = {255, 0, 0};

Color black = {0, 0, 0};

Square sq = {{44, 23}, 5, white, black};

**return** 0;

}

**3)**

**#include**<stdio.h>

**#include**<stdlib.h>

**typedef** **struct**{

**int** customertID;

**char** accountType;

**int** balance;

}BankCustomer;

**int** **main**(**void**){

BankCustomer customers[10];

FILE \*file;

**int** saving = 0;

**int** checking = 0;

**int** deposit = 0;

file = **fopen**("customersData.txt", "r");

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

**fscanf**(file, "%i", &customers[i].customertID);

**fscanf**(file, " %c", &customers[i].accountType);

**fscanf**(file, "%i", &customers[i].balance);

**printf**("Customer %i ID: %i. Account type is '%c' ", i+1, customers[i].customertID, customers[i].accountType);

**printf**("and balance is %i.\n", customers[i].balance);

**if**(customers[i].accountType == 's'){

saving += customers[i].balance;

}**else** **if**(customers[i].accountType == 'c'){

checking += customers[i].balance;

}**else**{

deposit += customers[i].balance;;

}

}

**printf**("\n");

**printf**("The average balance for all checking accounts: %i.\n", checking);

**printf**("The average balance for all saving accounts: %i.\n", saving);

**printf**("The average balance for all deposit accounts: %i.\n", deposit);

**fclose**(file);

**return** 0;

}

**Lesson 16**

**Prectice exercises**

**1)**

**#include**<stdio.h>

**#include** <math.h>

**double** **expApprox**(**double** x, **int** n) {

**double** exp;

**int** f = 1;

**for**(**int** j = 0; j <= n; j++){

**if**(j == 0){

f = 1;

}**else**{

f\*=j;

}

}

exp = **pow**(x, n)/f;

**return** exp;

}

**int** **main**() {

**int** N;

**double** x = 2.0, e = 0.0;

**printf**("How may terms?\n");

**scanf**("%i", &N);

**for**(**int** i = 0; i < N; i++){

e += expApprox(x, i);

}

**printf**("Approximation of an exponential function with %i terms is equal to: %f \n", N, e);

**return** 0;

}

**Lesson 17**

**Prectice exercises**

**1)**

**#include**<stdio.h>

**#include** <math.h>

**int** **smallestElem**(**double** vals[]) {

**double** min = vals[0];

**int** index = 0;

**for**(**int** i = 1; i < 4; i++){

**if**(vals[i] < min){

min = vals[i];

index = i;

}

}

**return** index;

}

**int** **main**() {

**int** ind = 0;

**double** arr[4];

**for**(**int** i = 0; i < 4; i++){

arr[i] = **rand**() % 50;

}

**for**(**int** i = 0; i < 4; i++){

**printf**("%f\t", arr[i]);

}

ind = smallestElem(arr);

**printf**("Index of the smallest element is: %i", ind);

**return** 0;

}

**Lesson 18**

**Prectice exercises**

**2)**

**#include**<stdio.h>

**int** **addSubarray**(**int** x[], **int** from, **int** to) {

**if**(from == to){

**return** x[to];

}**else**{

**return** addSubarray(x, from+1, to) + x[from];

}

}

**int** **main**() {

**int** arr[5] = {4, 5, 2, -1, 8};

**printf**("sum = %i", addSubarray(arr, 0, 4));

**return** 0;

}

**3)**

**#include**<stdio.h>

**double** **minValue**(**double** x[], **int** from, **int** to, **double** min) {

**if**(from == to && x[to] > min){

**return** min;

}**else** **if**(from == to && x[to] < min){

**return** x[to];

}**else**{

**if**(x[from] < min){

min = x[from];

**return** minValue(x, from+1, to, min);

}**else**{

**return** minValue(x, from+1, to, min);

}

}

}

**int** **main**() {

**double** arr[5] = {4.2, 5.5, 2.0, -1.9, 8.9};

**printf**("Minimum value in the array is %f.", minValue(arr, 0, 4, arr[4]));

**return** 0;

}

4)

**#include** <stdio.h>

**#include** <stdlib.h>

**void** **printBinary**(**int** n){

**if**(n>1){

**if**(n%2 == 0){

printBinary(n/2);

**printf**("%d", 0);

}**else**{

printBinary(n/2);

**printf**("%d", 1);

}

}**else**{

**printf**("%d", 1);

}

}

**int** **main**(**void**) {

**int** num;

**printf**("Enter a decimal number: ");

**scanf**("%d", &num);

printBinary(num);

**return** 0;

}

**Lesson 20**

**Prectice exercises**

**4)**

**#include**<stdio.h>

**#include**<stdlib.h>

**#include** <math.h>

**void** **concat**(**char** str1[], **char** str2[], **char** resultStr[]){

**int** i = 0, j = 0;

**while**(str1[j] != '\0'){

resultStr[j] = str1[j];

j++;

}

**while**(str2[i] != '\0'){

resultStr[j] = str2[i];

j++;

i++;

}

**for**(**int** z = 0; z < j; z++){

**printf**("%c", resultStr[z]);

}

}

**int** **main**() {

**char** arr1[10] = {'H', 'e', 'l', 'l', 'l', 'o', '!', '\0'};

**char** arr2[20] = {' ', 'H', 'o', 'w', ' ', 'a', 'r', 'e', ' ',

'y', 'o', 'u', '?', '\0'};

**char** arr[32];

concat(arr1, arr2, arr);

**return** 0;

}

**Lesson 21**

**Prectice exercises**

**1)**

**#include**<stdio.h>

**#include**<stdlib.h>

**int** **main**() {

FILE \*file;

**int** sum = 0;

file = **fopen**("values.txt", "r");

**if**(file == NULL){

**printf**("Problem opening file.\n");

**return** 1;

}

**char** val[32];

**int** n;

**do**{

**fscanf**(file, "%s", val);

sum += **atoi**(val);

}**while**(!**feof**(file));

**do**{

**fscanf**(file, "%i", n);

sum += n;

}**while**(!**feof**(file));

**printf**("Sum: %i", sum);

**fclose**(file);

**return** 0;

}

**Lesson 22**

**Prectice exercises**

**1)**

**#include**<stdio.h>

**int** **main**() {

**char** ch1 = 'X', ch2 = 'Y';

**char** ch3 = 'Z', ch4 = 'W';

**char** \*chp1 = &ch1;

**char** \*chp2;

\*chp1 = 'A';

chp2 = chp1;

ch3 = \*chp2;

chp2 = &ch4;

ch1 = 'B';

chp1 = &ch2;

**printf**("%c, %c, %c, %c\n", ch1, ch2, ch3, ch4);

**printf**("%p, %p, %c, %c", chp1, chp2, \*chp1, \*chp2);

**return** 0;

}

**2)**

**#include**<stdio.h>

**int** **main**() {

**int** a = 11, b = 20;

**int** x, y;

**int** \*ip1 = &a;

**int** \*ip2 = &x;

**int** \*\*ipp = &ip2;

\*ip2 = \*ip1 \* 7 + b;

ip1 = ip2;

ip2 = &y;

\*\*ipp = 88;

\*ipp = &b;

**printf**("%i, %i, %i, %i\n", a, b, x, y);

**printf**("%p, %p, %p\n", ip1, ip2, ipp);

**printf**("%i, %i, %p %i\n", \*ip1, \*ip2, \*ipp, \*\*ipp);

**return** 0;

}

**Lesson 24**

**Prectice exercises**

**1)**

**#include**<stdio.h>

**void** **writeln**(**char** letters[]) {

**int** i = 0;

**while** (letters[i] != '\0') {

**putchar**(letters[i]);

i++;

}

**putchar**('\n');

}

**int** **main**( ) {

**char** arr[] = "abcdef";

**char** ptr[] = "ghijklmnop";

writeln(arr);

writeln(ptr);

writeln("qrstuv");

**return** 0;

}

**2)**

**#include**<stdio.h>

**void** **stringCopy**(**char** \*fromStr, **char** \*toStr) {

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

\*toStr = \*fromStr;

fromStr++;

toStr++;

}

}

**\_Bool** **areEqual**(**char** \*str1, **char** \*str2) {

**while**(\*str1 != '\0' || \*str2 != '\0'){

**if**(\*str1 != \*str2){

**return** 0;

}

str1++;

str2++;

}

**return** 1;

}

**void** **reverse**(**char** \*fromStr, **char** \*toStr) {

**int** n = 0;

**char** \*temp = fromStr;

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

fromStr++;

n++;

}

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

\*toStr = \*(temp+i);

toStr++;

}

}

**void** **concatenate**(**char** \*str1, **char** \*str2, **char** \*resultStr){

**int** i = 0, j = 0;

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

\*(resultStr + i) = \*str1;

str1++;

i++;

}

j = i;

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

\*(resultStr+j) = \*str2;

str2++;

j++;

}

}

**int** **main**() {

**char** word1[] = "Hello";

**char** word2[] = "Hi there";

**char** word3[] = {'H', 'e', 'l', 'l', 'o', '\0', 'X'};

**char** word4[] = "Hi";

**char** result[50];

// Test code for stringCopy

/\* stringCopy(word1, result);

printf("Test 1.1 result: %s\n", result);

stringCopy(word2, result);

printf("Test 1.2 result: %s\n", result);

stringCopy(word3, result);

printf("Test 1.3 result: %s\n", result);

// Test code for areEqual

printf("Test 2.1 result: %i\n", areEqual(word1, word2));

printf("Test 2.2 result: %i\n", areEqual(word1, word3));

printf("Test 2.3 result: %i\n", areEqual(word2, word4));

printf("Test 2.4 result: %i\n", areEqual(word4, word2));

\*/

// Test code for reverse

reverse(word1, result);

**printf**("Test 3.1 result: %s\n", result);

reverse(word2, result);

**printf**("Test 3.2 result: %s\n", result);

reverse(word3, result);

**printf**("Test 3.3 result: %s\n", result);

/\* // Test code for concatenate

concatenate(word1, word2, result);

printf("Test 4.1 result: %s\n", result);

concatenate(word3, word4, result);

printf("Test 4.2 result: %s\n", result);

concatenate(word1, word1, result);

printf("Test 4.3 result: %s\n", result);

\*/

**return** 0;

}

**Lesson 25**

**Prectice exercises**

**1)**

**#include**<stdio.h>

**#include**<stdlib.h>

**char**\* **stringCopy**(**char** \*fromStr){

**char** \*toStr = (**char**\*)**malloc**(**sizeof**(**char**));

**char** \*res = toStr;

**if**(toStr == NULL){

**return** 1;

}

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

\*toStr = \*fromStr;

fromStr++;

toStr++;

}

**return** res;

}

**int** **main**() {

**char** word1[] = "Hello";

**char** word2[] = "Hi there";

**char** word3[] = {'H', 'e', 'l', 'l', 'o', '\0', 'X'};

//char \*result = stringCopy(word1);

**char** \*result = stringCopy(word3);

**printf**("%s\n", result);

/\*while(\*result != '\0'){

putchar(\*result);

result++;

}\*/

**return** 0;

}

**2)**

**#include**<stdio.h>

**#include**<stdlib.h>

**typedef** **struct**{

**int** rainAmount;

**double** temperature;

**double** windSpeed;

}weather;

**int** **totalRainfall**(weather \*r){

**int** total = 0;

**for**(**int** i = 0; i < 12; i++){

total += (r+i)->rainAmount;

}

**return** total;

}

**double** **lowestAverTemp**(weather \*t){

**double** min = 100;

**for**(**int** i = 0; i < 12; i++){

**if**((t+i)->temperature < min){

min = (t+i)->temperature;

}

}

**return** min;

}

**double** **averWindSp**(weather \*s){

**double** result = 0;

**for**(**int** i = 0; i < 12; i++){

result += (s+i)->windSpeed;

}

result = result/12;

**return** result;

}

**int** **main**() {

weather \*monthlyData = (weather\*)**malloc**(12\***sizeof**(weather));

FILE \*file = **fopen**("astana.txt", "r");

**if**(file == NULL){

**printf**("Problem opening file");

**return** 1;

}

**printf**("Rain(cm)\t\tAvg.Temp(C)\tAvg. Wind(m/s)\n");

**for**(**int** i = 0; i < 12; i++){

**fscanf**(file, "%i", &(monthlyData+i)->rainAmount);

**fscanf**(file, "%lf", &(monthlyData+i)->temperature);

**fscanf**(file, "%lf", &(monthlyData+i)->windSpeed);

**printf**("%i%20.1f%20.1f\n", (monthlyData+i)->rainAmount, (monthlyData+i)->temperature, (monthlyData+i)->windSpeed);

}

**printf**("\nThe total rainfall for the year: %i.\n", totalRainfall(monthlyData));

**printf**("\nThe lowest average temperature: %.1f.\n", lowestAverTemp(monthlyData));

**printf**("\nThe average wind speed for the year: %.1f.\n", averWindSp(monthlyData));

**fclose**(file);

**free**(monthlyData);

**return** 0;

}

**Lesson 26**

**#include**<stdio.h>

**#include**<stdlib.h>

**typedef** **struct** \_doubleNode{

**double** value;

**struct** \_doubleNode\* link;

}doubleNode;

**void** **printList**(doubleNode \*node){

**if**(node != NULL){

**printf**("%.1f ", node->value);

printList(node->link);

}

}

**int** **main**() {

doubleNode\* first = (doubleNode\*)**malloc**(**sizeof**(doubleNode));

first->value = 12.3;

first->link = **malloc**(**sizeof**(doubleNode));

first->link->value = 22.7;

first->link->link = **malloc**(**sizeof**(doubleNode));

first->link->link->value = 8.7;

first->link->link->link = **malloc**(**sizeof**(doubleNode));

first->link->link->link->value = 55.5;

first->link->link->link->link = NULL;

printList(first);

**return** 0;

}

**Reading input from the user**

**#include**<stdio.h>

**#include**<stdlib.h>

**typedef** **struct** \_doubleNode{

**double** value;

**struct** \_doubleNode\* link;

}doubleNode;

**void** **printList**(doubleNode \*node){

**if**(node != NULL){

**printf**("%.1f ", node->value);

printList(node->link);

}

}

**void** **freeAll**(doubleNode \*node) {

**if** (node != NULL) {

freeAll(node->link);

**printf**("Freeing up %f \n", node->value);

**free**(node);

}

}

**int** **main**() {

doubleNode\* first = NULL;

doubleNode\* prev = NULL;

**do**{

**printf**("Enter in a double: ");

**double** x;

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

**if**(x < 0.0){

**break**;

}

doubleNode \*newNode = **malloc**(**sizeof**(doubleNode));

newNode->value = x;

newNode->link = NULL;

**if**(first == NULL){

first = newNode;

}**else**{

prev->link = newNode;

}

prev = newNode;

}**while**(1);

printList(first);

**printf**("Done.");

**freeAll**(first);

**return** 0;

}

**Prectice exercises**

**1)**

**#include**<stdio.h>

**#include**<stdlib.h>

**typedef** **struct** \_charNode{

**char** value;

**struct** \_charNode\* link;

}charNode;

**void** **printList**(charNode \*node){

**if**(node != NULL){

**printf**("%c ", node->value);

printList(node->link);

}

}

**void** **printInReverse**(charNode \*node){

**if**(node != NULL){

printInReverse(node->link);

**printf**("%c ", node->value);

}

}

**void** **freeAll**(charNode \*node) {

**if** (node != NULL) {

freeAll(node->link);

**printf**("\nFreeing up %c \n", node->value);

**free**(node);

}

}

**int** **main**(){

charNode\* first = NULL;

charNode\* prev = NULL;

**char** ch;

**for**(ch = 'a'; ch <= 'z'; ch++){

charNode \*newNode = **malloc**(**sizeof**(charNode));

newNode->value = ch;

newNode->link = NULL;

**if**(first == NULL){

first = newNode;

}**else**{

prev->link = newNode;

}

prev = newNode;

}

printList(first);

**printf**("\n");

printInReverse(first);

**printf**("Done.");

freeAll(first);

**return** 0;

}

**2)**

**#include**<stdio.h>

**#include**<stdlib.h>

**typedef** **struct** \_weather{

**int** rainAmount;

**double** temperature;

**double** windSpeed;

**struct** \_weather \*link;

}weather;

**void** **printAll**(weather \*node){

**if** (node != NULL) {

**printf**("%i%20.1f%20.1f\n", node->rainAmount, node->temperature, node->windSpeed);

printAll(node->link);

}

}

**void** **freeAll**(weather \*node){

**if** (node != NULL){

freeAll(node->link);

**printf**("%i%20.1f%20.1f\n", node->rainAmount, node->temperature, node->windSpeed);

**free**(node);

}

}

**int** **totalRainfall**(weather \*node){

**if**(node == NULL){

**return** 0;

}**else**{

**return** node->rainAmount + totalRainfall(node->link);

}

}

**double** **lowestAverTemp**(weather \*node, **double** t){

**if**(node == NULL){

**if**(node->temperature < t){

**return** node->temperature;

}**else**{

**return** t;

}

}**else**{

**if**(node->temperature < t){

**return** node->temperature;

}

**return** lowestAverTemp(node->link, t);

}

}

**double** **averWindSp**(weather \*node){

**if**(node == NULL){

**return** 0.0;

}**else**{

**return** node->windSpeed + averWindSp(node->link);

}

}

**int** **main**() {

weather \*first = NULL;

weather \*prev = NULL;

FILE \*file = **fopen**("astana.txt", "r");

**if**(file == NULL){

**printf**("Problem opening file");

**return** 1;

}

**printf**("Rain(cm)\t\tAvg.Temp(C)\tAvg. Wind(m/s)\n");

printAll(first);

**for**(**int** i = 0; i < 12; i++){

weather \*newNode = (weather\*)**malloc**(12\***sizeof**(weather));

**fscanf**(file, "%i", &newNode->rainAmount);

**fscanf**(file, "%lf", &newNode->temperature);

**fscanf**(file, "%lf", &newNode->windSpeed);

**if**(first == NULL){

first = newNode;

}**else**{

prev->link = newNode;

}

prev = newNode;

}

printAll(first);

**printf**("\nThe total rainfall for the year: %i.\n", totalRainfall(first));

**printf**("\nThe lowest average temperature: %.1f.\n", lowestAverTemp(first, 100.0));

**printf**("\nThe average wind speed for the year: %.1f.\n", averWindSp(first)/12);

**fclose**(file);

freeAll(first);

**return** 0;

}

**Lesson 26**

**#include**<stdio.h>

**#include**<stdlib.h>

**int** **findInt**(**int** x[], **int** size, **int** key){

**for**(**int** i = 0; i < size; i++){

**if**(x[i] == key){

**return** i;

}

}

**return** -1;

}

**int** **binarySearch**(**int** x[], **int** size, **int** key){

**int** min = 0;

**int** max = size-1;

**while**(min <= max){

**int** guess = (min + max)/2;

**if**(x[guess] == key){

**return** guess;

}**else** **if**(x[guess] < key){

min = guess + 1;

}**else**{

max = guess - 1;

}

}

**return** -1;

}

**int** **main**(){

**int** vals1[8] = {1, 2, 3, 4, 5, 6, 7, 8};

**int** vals2[8] = {1, 2, 3, 4, 6, 7, 8, 9};

**int** vals3[10] = {3, 5, 1, 3, -2, 33, 2, 5, 9, 0};

**int** vals4[1] = {42};

**printf**("Test1.1: %i\n", binarySearch(vals1, 8, 8));

**printf**("Test1.2: %i\n", binarySearch(vals1, 8, 5));

**printf**("Test2.1: %i\n", binarySearch(vals2, 8, 5));

**printf**("Test2.2: %i\n", binarySearch(vals2, 8, 3));

**printf**("Test2.3: %i\n", binarySearch(vals2, 8, 33));

**printf**("Test3.1: %i\n", binarySearch(vals3, 10, 5));

**printf**("Test3.2: %i\n", binarySearch(vals3, 10, -2));

**printf**("Test3.3: %i\n", binarySearch(vals3, 10, -3));

**printf**("Test4.1: %i\n", binarySearch(vals4, 1, 42));

**printf**("Test4.2: %i\n", binarySearch(vals4, 1, 41));

**return** 0;

}

**Prectice exercises**

**2)**

**#include**<stdio.h>

**#include**<stdlib.h>

**int** **func**(**double** x){

**return** -2 \* x \* x \* x - x + 2;

}

**double** **zeroFinder**(**double** min, **double** max, **double** error){

**while**(max - min > error){

**double** guess = (max + min)/2;

**if**(func(guess) == 0){

**return** guess;

}**else** **if**(func(guess) > 0){

min = guess;

}**else**{

max = guess;

}

}

**return** min;

}

**double** **zeroFindRec**(**double** min, **double** max, **double** error){

**if**(max - min > error){

**double** guess = (max + min)/2;

**if**(func(guess) == 0){

**return** guess;

}**else** **if**(func(guess) > 0){

**return** zeroFindRec(guess, max, error);

}**else**{

**return** zeroFindRec(min, guess, error);

}

}

**return** min;

}

**int** **main**(){

**printf**("zero: %f \n", zeroFindRec(0.0, 1.0, 0.01));

**printf**("zero: %f \n", zeroFinder(0.0, 1.0, 0.001));

**printf**("zero: %f \n", zeroFindRec(0.0, 1.0, 0.000001));

**printf**("zero: %f \n", zeroFinder(0.0, 1.0, 0.0000000001));

**return** 0;

}

**3)**

**#include**<stdio.h>

**#include**<stdlib.h>

**int** **func**(**double** x){

**return** x \* x \* x + 2 \* x \* x - 2;

}

**double** **zeroFinder**(**double** min, **double** max, **double** error){

**while**(max - min > error){

**double** guess = (max + min)/2;

**if**(func(guess) == 0){

**return** guess;

}**else** **if**(func(guess) > 0){

min = guess;

}**else**{

max = guess;

}

}

**return** min;

}

**double** **zeroFindRec**(**double** min, **double** max, **double** error){

**if**(max - min > error){

**double** guess = (max + min)/2;

**if**(func(guess) == 0){

**return** guess;

}**else** **if**(func(guess) > 0){

**return** zeroFindRec(guess, max, error);

}**else**{

**return** zeroFindRec(min, guess, error);

}

}

**return** min;

}

**int** **main**(){

**printf**("zero: %f \n", zeroFindRec(-1.33, 0.0, 0.01));

**printf**("zero: %f \n", zeroFinder(-1.33, 0.0, 0.001));

**printf**("zero: %f \n", zeroFindRec(0.0, 1.0, 0.000001));

**printf**("zero: %f \n", zeroFinder(0.0, 1.0, 0.0000000001));

**return** 0;

}