100 C Language Tasks with Solutions

by ADAM HIGHERSTEIN

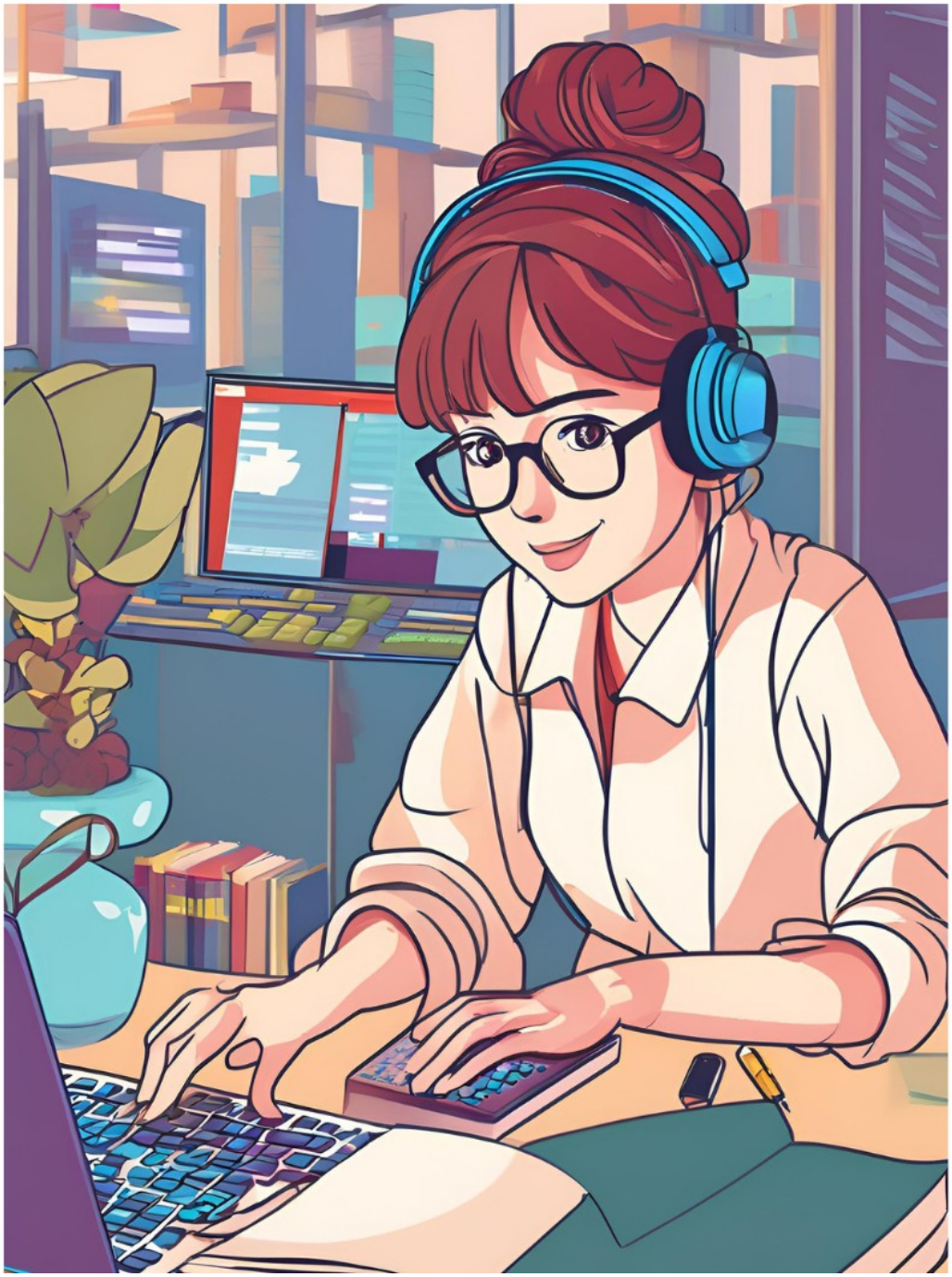


Table of Contents

Programming Exercises with Solutions!

Learning by doing!

Language now: C

Free book
This is the first version

Pls, give comments, feedback, new ideas to the 2. version!!

Comments can be sent to:
darry.robinson@gmail.com

Thank You!

Introduction

Try to do tasks first yourself without checking solutions!

Ask if you have problems.

And finally check right solutions!

Check also author's YT Channel:
https://www.youtube.com/@adamhigherstein8986/videos

Tool that we use in this tutorial:
DevC++

It can be downloaded from
https://bloodshed.net/

SET 1: Tool & basics

Topics:
Installation of programming tool
Testing installation with ”Hello world!” classic code.
Variables and datatypes
Printing

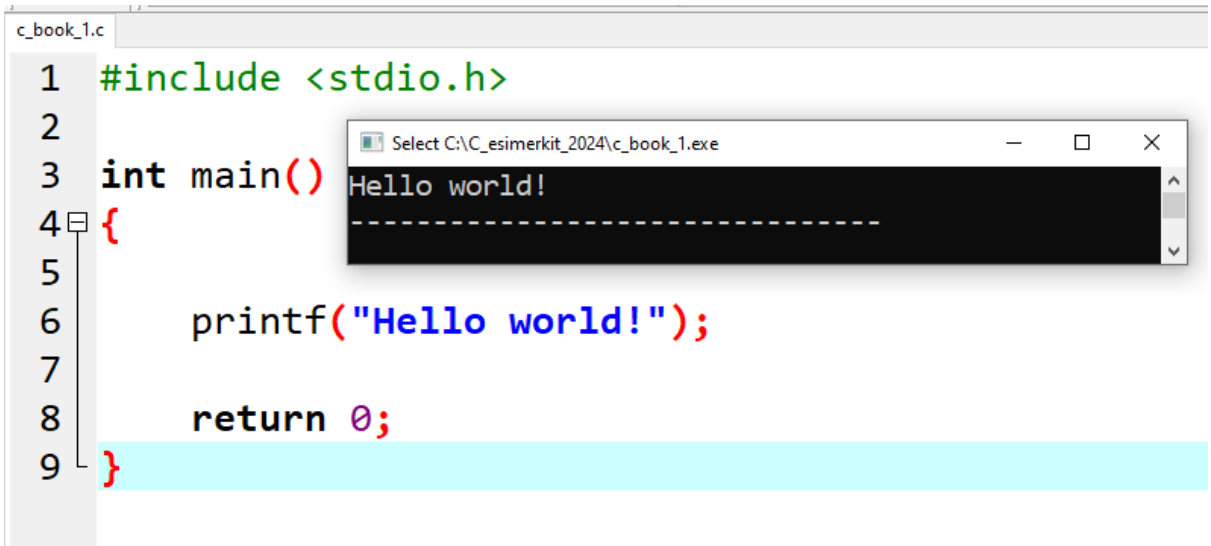
Task 1

Install DevC++ first.

Type this code, save the file using extension .c and run the program

```
#include <stdio.h>

int main()
{

    printf("Hello world!");

    return 0;
}
```

This is the output:



Task 2

Define suitable variables for these values:

a) 999999
b) 5.55555555555
c) 'x'
e) 2.33
f) 10
g) 300
h) 9 billions
i) 3 billions

Solution

```
int t1 = 999999;
double t2 = 5.55555555555;
char t3 = 'x';
float t4 = 2.33;
short t5 = 10;
unsigned  short t6 = 300;
float t7 = 9000000000;
```

OR

```
longlong t7 = 9000000000;
unsigned int t8 = 3000000000;
```

OR

```
longlong t8 = 3000000000;
```

Task 3

Our programs uses Ohm's law to calculate the resistance.
Voltage and current are given.

Solution

```
float U = 50;
float I = 20;
float R = U/I;
printf("R on %f \n", R);
```

Test run

```
float U = 50;
float I = 20;
float R = U/I;
printf("R on %f \n", R);
```

```
C:\C_esimerkit_2024\c_book_1.exe
R on 2.500000
```

Task 4

User gives the speed of the car (km/h) and the distance (km). Program calculates amount of time.

a) in hours
b) in whole hours and minutes

Solution

a)

```
float v = 75;
float s = 1150;
float t = s/v;
printf("It takes %f hours \n", t);
```

Test run

```
float v = 75;
float s = 1150;
float t = s/v;
printf("It takes %f hours \n", t);
```

```
C:\C_esimerkit_2024\c_book_1.exe
It takes 15.333333 hours
```

b)

```
int whole_hours = (int) t;
int minutes = (int) ( (t - whole_hours) * 60);
printf("It takes %d hours and %d minutes \n", whole_hours,
minutes);
```

Test run

```
float v = 75;
float s = 1150;
float t = s/v;
int whole_hours = (int) t;
int minutes = (int) ( (t - whole_hours) * 60);
printf("It takes %d hours and %d minutes \n", whole_hours, minutes);
```

```
C:\C_esimerkit_2024\c_book_1.exe
It takes 15 hours and 19 minutes
```

Task 5

Our program calculates BMI. Weight and height are given.

Solution

```
float height_cm = 200;  // cm
float weight = 100;  // kg

float height_m = height_cm/100;
float bmi = weight/(height_m * height_m);

printf("bmi is %f \n",  bmi);
```

Test run

```
float height_cm = 200;  // cm
float weight = 100;  // kg
float height_m = height_cm/100;
float bmi = weight/(height_m * height_m);

printf("bmi is %f \n",  bmi);
```

```
C:\C_esimerkit_2024\c_book_1.exe
bmi is 25.000000
```

Task 6

Create a euro converter: dollars to euros.

Solution

we can take the current exchange value now:

1 USD to EUR - US Dollars to Euros Exchange Rate

+4.48%. (1Y). **1 USD = 0.959589 EUR**. Nov 23, 2024, 17: ...

```
float dollars = 200;
float current_coeff = 0.9595;
float euros = current_coeff * dollars;
printf("sum is %f \n",  euros);
```

Task 7

Convert seconds to hours, minutes, seconds.

Solution

```
int allSeconds = 123456;
int hours = allSeconds / 3600;  // hours is  34
allSeconds = allSeconds - hours * 3600;  // allSeconds is  1056
int minutes = allSeconds/60;  // minutes 17
int seconds = allSeconds - minutes *  60; // 36 seconds
printf("Hours: %d, minutes: %d and seconds: %d", hours,
minutes, seconds);
```

Task 8

Convert euros to 5, 10, 20, 50, 100, 200, 500 euros bills.

Solution

```
int euros = 1234;
int b500 = euros/500;  // 2
euros = euros - b500*500; // 234
int b200 = euros/200;  // 1
euros = euros - b200*200; // 34
int b100 = euros/100;  // 0
euros = euros - 100*100; // 34
int b50 = euros/50;  // 0
euros = euros - b50 * 50; // 34
int b20 = euros/20; // 1
euros = euros - b20*20; // 14
int b10 = euros/10;  // 1
int b5 = euros - b10*10; // 0
int remainingEuros = euros - b5*5;  // 4
```

SET 2: Decision making

Topics:
decision making, branching, if else

Task 9

User gives a value and our program tells if the value is > 100 or not.

```
int x;
printf("Give a whole number: \n");
scanf("%d", &x);

if (x > 100)
   printf("It is over 100 \n");
else
  printf("It is not over 100 \n");
```

Task 10

Write a program that reads two integer values.
If the first is less than the second, print the message "up".
If the second is less than the first, print the message "down ".
If the numbers are equal, print the message "equal".

Solution

```
int a, b;
printf("Give a 2 whole numbers: \n");
printf("Give a 1. whole number: \n");
scanf("%d", &a);
printf("Give a 2. whole number: \n");
scanf("%d", &b);
if (a < b)
 printf("up \n");
else if (a > b)
 printf("down \n");
else
 printf("equal \n");
```

Task 11

User enters a weekday number and the program tells the name of the day in Germany.

Solution

```
int nro;
printf("Give the weekday nr (Monday = 1) \n");
scanf("%d", &nro);

if (nro == 1)
  printf("Montag \n");
else if (nro == 2)
  printf("Dienstag \n");
else if (nro == 3)
  printf("Mittwoch \n");
else if (nro == 4)
  printf("Donnerstag \n");
else if (nro == 5)
  printf("Freitag \n");
else if (nro == 6)
  printf("Samstag \n");
else if (nro == 7)
  printf("Sonntag \n");
else
  printf("Not a suitable nr \n");
```

Task 12

Program solves a quadratic equation
Note: you have to include math.h to your source file and then use sqrt() function.

Solution

```
float a, b, c;
float x1, x2;
float diskr;
printf("Give coifficients a, b and c: \n");
printf("Give a: \n");
scanf("%f", &a);
printf("Give b: \n");
scanf("%f", &b);
printf("Give c: \n");
scanf("%f", &c);
diskr = b*b - 4 * a * c;
if (diskr < 0)
 printf("No real roots \n");
else
{
 x1 = (-b + sqrt(diskr))/(2*a);
 x2 = (-b - sqrt(diskr))/(2*a);
 printf("x1 = %d \n", x1);
 printf("x2 = %d \n", x2);
}
```

Task 13

User gives a month number and our program tells the number of days in that month.

Solution

```
int kk;
printf(Give the month number (1 - 12) \n");
scanf("%d", &kk);

if (kk == 4 || kk == 6 || kk == 9 || kk == 11)
 printf("30 \n");
else if (kk == 2)
 printf("28/29 \n");
else
 printf("31 \n");
```

Task 14

User gives the lengths of the triangle's sides. Program tells what is the triangle like and calculates the area of the triangle

We may have these types
Equilateral triangle

Isosceles triangle

Right angled triangle

Normal triangle

Solution

```
        float a, b, c;
printf("Give the lengths of the sides \n");
scanf("%f", &a);
scanf("%f", &b);
scanf("%f", &c);

if (a == b && a == c)
   printf("Equilateral triangle \n");
       else if (a == b || a == c || b == c)
        printf("Isosceles triangle \n");
       else if (a*a + b*b == c*c  || a*a + c*c == b*b || b*b +
         c*c == a*a )
         printf("Right angled triangle \n");
       else
        printf("Basic triangle \n");

float s = (a + b + c)/2;
float tempvalue = s*(s-a)*(s-b)*(s-c);
float area = sqrt(tempvalue);

printf("Area is %f \n", area);

// Heron's formula is used for the area
// area =  SQRT(s*(s-a)*(s-b)*(s-c))
// s = (a + b + c)/2
```

Task 15

Create a program: what is the biggest of 3 given values?

Solution

```
  // Method 1
int p1 = 4; int p2 = 6; int p3 = 8;
if (p1 > p2)
  if (p2 > p3)
     printf("Biggest is %d \n", p1);
  else
       if (p3 > p1)
```

```
        printf(""Biggest is %d \n", p3);
        else
        printf(""Biggest is %d \n", p1);
    else
    if (p2 > p3)
        printf(""Biggest is %d \n", p2);
    else
        printf(""Biggest is %d \n", p3);
```

// Method 2

```
    if (p1 > p2 && p2 > p3)
       printf(""Biggest is %d \n", p1);
    else if (p2 > p1 && p2 > p3)
               printf(""Biggest is on %d \n", p2);
            else
              printf(""Biggest is on %d \n", p3);
```

// Method 3

```
  int biggest = p1;
  if (p2 > biggest)
     biggest = p2;
  if (p3 > biggest)
     biggest= p3;
   printf(""Biggest is  %d \n", biggest);
```

SET 3: Loops

Topics:
Loops: for, while, do while

Task 16
Program calculates the sum of values 1 - 5.
Use: for, while and do-while

Solution

```
// for
                    int sum = 0;
int p;
for (p = 1; p <= 5; p++)
{
      sum += p;
}

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

// while
sum = 0;
p = 1;
while (p <= 5)
{
      sum += p;
      p++;
}

printf("sum on %d \n", sum);

// do while
sum = 0;
p = 1;
do
{
      sum += p;
      p++;
}
while (p <= 5);

printf("sum on %d \n", sum);
```

Task 17

Program calculates the sum of even numbers between 2 - 40.
Use: for, while and do-while

Solution

```
// for
                    int sum = 0;
 int p;
 for (p = 2; p <= 40; p += 2)          // p = p + 2;
 {
       sum += p;
 }

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

// while
 sum = 0;
 p = 2;
 while (p <= 40)
 {
       sum += p;
       p += 2;
 }

 printf("sum on %d \n", sum);

// do while
 sum = 0;
 p = 2;
 do
 {
       sum += p;
       p += 2;
 }
 while (p <= 40);

 printf("sum on %d \n", sum);
```

Task 18

Program calculates sum: 5, 10, 15, .. 100.
Use: for, while and do-while

Solution

```
//      for
                    int sum = 5;
 int p;
 for (p = 5; p <= 100; p += 5)          // p = p + 5;
```

```
 {
       sum += p;
 }

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

// while
 sum = 5;
 p = 5;
 while (p <= 100)
 {
       sum += p;
       p += 5;
 }

 printf("sum on %d \n", sum);

// do while
 sum = 5;
 p = 5;
 do
 {
       sum += p;
       p += 5;
 }
 while (p <= 100);

 printf("sum on %d \n", sum);
```

Task 19

Program generates 50 random numbers (between 1 to 10) and calculates sum and average.

Solution

```
int sum = 0;
int i;
for (i = 0; i < 50; i++)
{
 sum = sum + rand() % 10 + 1;
}

printf("sum is  %d \n", sum);
float aver = (float) sum/50;
printf("average is %f \n", aver);
```

Task 20

Program throws dice 100 times and tells amounts of different values (1, 2, 3, 4, 5, and 6).

Solution

```
        // reset random number generator
  rand(time(NULL));


int n1 = 0;  int n2 = 0;  int n3 = 0;
int n4 = 0;  int n5 = 0;  int n6 = 0;

int i;
for (i = 0; i < 10000; i++)
{
    int x = rand() % 6 + 1;

    switch (x)
    {
         case 1: n1++; break;
         case 2: n2++; break;
         case 3: n3++; break;
         case 4: n4++; break;
         case 5: n5++; break;
         case 6: n6++; break;
    }
}

 printf("1: %d \n", n1);
 printf("2: %d \n", n2);
 printf("3: %d \n", n3);
 printf("4: %d \n", n4);
 printf("5: %d \n", n5);
 printf("6: %d \n", n6);
```

Task 21

Create an account manager with menu:
User can make deposits
Do withdrawal
Check the balance

Create a menu
take money
add money
check balance
exit

Solution

```
  int saldo = 999;
```

```
while (1)
{
 system("cls");
 printf("Menu \n");
 printf("1 ==> Take money \n");
 printf("2 ==> Add money \n");
 printf("3 ==> Check balance \n");
 printf("0 ==> Lopeta\n");

 int v = 9;
 printf("Your choice?\n");
 scanf("%d", &v);

 if (v == 1)
 {
       int sum;
       printf("Give the sume: \n");
       scanf("%d", &sum);
       if (sum <= saldo)
       {
           saldo -= sum;
           printf("Balance is now %d \n", saldo);
       }
       else
       {
           printf("Not enough money \n");
     printf("Push any key to go on...\n");
            getchar(); getchar();
        }
 }

 if (v == 2)
 {
       int summa;
       printf("Give the sum: \n");
       scanf("%d", &summa);
     saldo += summa;
     printf("Balance is %d \n", saldo);
       printf("Push any key to go on...\n");
       getchar();getchar();
}

if (v == 3)
 {
       printf("Balance is %d \n", saldo);
       printf("Push any key to go on...\n");
```

```
        getchar();getchar();
       }

    if (v == 0)
     {
           break;
       }

    }
```

Note:
Variable for account balance has to be global!
=> declare it outside (above) the while loop
When user takes money you have to check if there is enough money...

Task 22

Try to solve this equation:
3x^3 - 4x^2 + 9x +5 = 0
Here ^ means exponent

Solution

```
       double x, y;

        for (x = -5; x < 5; x += 0.0001)
        {
              y = 3*x*x*x  - 4*x*x + 9*x + 5;
              if (y > -0.001 && y < 0.001)
                break;
        }

        printf("%f \n", x);
        printf("%f \n", y);
```

Task 23

Print this kind shape: character and amount of rows are given.
o
oo
ooo
oooo
ooooo
oooooo

and so on.

Solution

```
    char merkki = 'x';
    int rivit = 20;
    int i;
    int j;

    for (i = 1; i <= rivit; i++)
    {
     for (j = 0; j < i; j++)
        printf("%c", merkki);

     printf("\n");
     }
```

Task 24

Create this kind of shape: amount is given by the user

o
oo
ooo
oooo
ooooo
oooooo
ooooooo
oooooooo
ooooooooo
oooooooo
ooooooo
oooooo
ooooo
oooo
ooo
oo
o

Solution

```
int main()
{
    printf("How many rows max? \n");
    int n;
    scanf("%d", &n);

    int i, j;
    for (i = 0; i < n; i++)
    {
      for (j = 0; j <= i; j++)
      printf("%c", 'o');
```

```
      printf("\n");

     }
     for (i = n; i >= 0; i--)
     {
        for (j = 0; j <= i; j++)
       printf("%c", 'o');
      printf("\n");
    }
     return 0;
}
```

Task 25

Generate a lotto row.
Rules: Select seven numbers from 1 to 40

Solution

Method 1 (funny way, a lot of computer work :))

```
srand(time(NULL));
int n1 = 0, n2 = 0, n3 = 0, n4 = 0, n5 = 0, n6 = 0, n7 = 0;

while (1)
{
 n1 = rand() % 40 + 1;
  n2 = rand() % 40 + 1;
  n3 = rand() % 40 + 1;
  n4 = rand() % 40 + 1;
  n5 = rand() % 40 + 1;
  n6 = rand() % 40 + 1;
  n7 = rand() % 40 + 1;

  if (n1 != n2 && n1 != n2 && n1 != n3 && n1 != n4 && n1 != n5
  && n1 != n6 && n1 != n7 && n2 != n3   && n2 != n4  && n2 != n5
     && n2 != n6   && n2 != n7
     && n3 != n4  && n3 != n5  && n3 != n6 && n3 != n7
     && n4 != n5  && n4 != n6  && n4 != n7
      && n5 != n6   && n5 != n7
        && n6 != n7)
        break;

}
```

```
printf("%d %d %d %d %d %d %d", n1, n2, n3, n4, n5, n6, n7);
```

Way 2

```
  int nros[]  = {0,0,0,0,0,0,0};

   int i;
   for (i = 0; i < 7; i++)
   {

      int existed_already = 0;
      int newnr = rand() % 40 + 1;
      int j;
      for (j = 0; j <= i; j++)
      {
           if (nros[j] == newnr)
           {
                 existed_already = 1;
                 break;
         }
       }

       if (existed_already == 1)
         i--;
       else
         nros[i] = newnr;

   }
```

Task 26

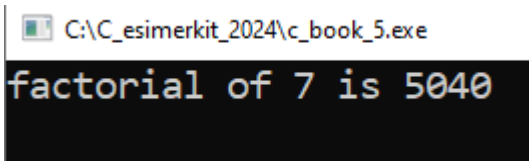
Calculate factorial and amount of combinations.

Solutions

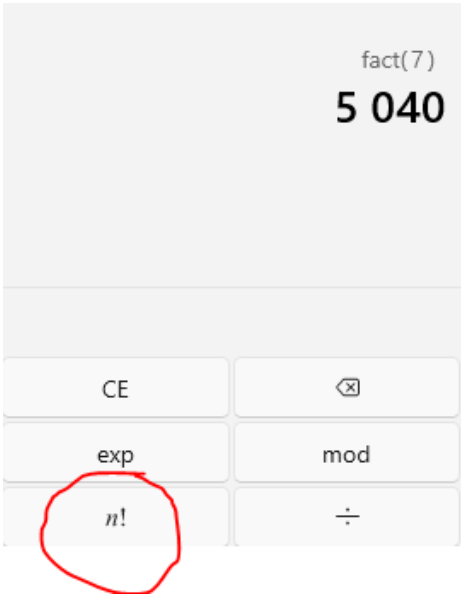
Factorial

```
    // factorial
    /*
    0! = 1
    n! = 1 * 2 * ... * n
    */

    int f = 1;
    int i;
    // now  factorial of 7
    int n = 7;
    if (n == 0)
      f = 1;
```

```
else
for (i = 1; i <= 7; i++)
{
 f = f * i;
}

printf("factorial of %d is %d \n", n, f);
```



Calculator check:



Task 27

Create a program that calculates amount of combinations

Combinations theory first:

// combinations
// amount = n!/k*(n-k)!
// n is the whole population
// k is the sample

Solution

```
int n, k, amount;

n = 5;
k = 3;

int i;

   int n_fact = 1;

for (i = 1; i <= n; i++)
  n_fact = n_fact * i;

  int  k_fact = 1;

for (i = 1; i <= k; i++)
  k_fact = k_fact * i;


   int diff_fact = 1;

for (i = 1; i <= n-k; i++)
  diff_fact = diff_fact * i;

amount = n_fact/(k_fact*diff_fact);

printf("amount is %d \n", amount);
```

/*
illustration
the whole population could be "abcde", so n is 5
samples are then

abc
abd
abe

acd
ace
ade

bcd

bde
bce

cde

answer is 10 combinations
*/

SET 4: Arrays

Task 28
Create a program that
a) fills an array with random numbers
b) prints an array
c) calculates the sum
d) calculates the average
e) finds the max and min
f) finds a spesific values
g) tells how many times some value exists in an array

Solution

```
int vals[5];
int i;
for (i = 0; i < 5; i++)
{
 vals[i] = rand();
}


int sum = 0;
int i;
for (i = 0; i < 5; i++)
{
 sum += vals[i];
}

printf("Sum is %d \n", sum);

// printing
for (i = 0; i < 5; i++)
{
 printf("%d \n",vals[i]);
```

```
}


// minimum

int min = vals[0];
for (i = 0; i < 5; i++)
{
 if (vals[i] < min)
    min = vals[i];
}

printf("smallest is %d \n", min);

// maximum

int max = luvut[0];
for (i = 0; i < 5; i++)
{
 if (vals[i] > max)
       max = vals[i];
}
printf("Biggest is: %d \n", max);

// search for a value
// we put there some value that we then know it exits
vals[3] = 99999;
luvut[3] = 12345;
int x = 99999;
int result = -1;
for (i = 0; i < 5; i++)
{
 if (x == vals[i])
 {
       result = i;
       break;
 }
}

if (result == -1)
  printf("Not found:( \n ");
else
  printf("Found, location is %d\n", result);
```

Task 29

Create a program that multiplies array values with given value.

Solution

```
int a[4];
a[0] = 10; a[1] = -5; a[2] = 30; a[3] = 99;
int i;
// original
for (i = 0; i < 4; i++)
       printf("%d \n", a[i]);

for (i = 0; i < 4; i++)
  a[i] = 2 * a[i];

// manipulated
for (i = 0; i < 4; i++)
  printf("%d \n", a[i]);
```

Task 30

Create a program that
calculates the sum of 2 array values to 3. array.

Solution

```
int a[4];
a[0] = 10; a[1] = -5; a[2] = 30; a[3] = 99;
int b[4];
b[0] = 66; b[1] = 33; b[2] = 0; b[3] = -110;

int c[4];

int i;
for (i = 0; i < 4; i++)
 c[i] = a[i] + b[i];


for (i = 0; i < 4; i++)
          printf("%d \n", c[i]);
```

Task 31

Create a program that fills and prints a 3x4 array,

Solution

```
int matr[3][4];

int i, j;
```

```
 for (i = 0; i < 3; i++)
 for (j = 0; j < 4; j++)
        matr[i][j] = rand() % 100; // now values 0 - 99 added

 // basic output
 for (i = 0; i < 3; i++)
 for (j = 0; j < 4; j++)
        printf("%d \n", matr[i][j]);

// arraylike output
for (i = 0; i < 3; i++)
{
 for (j = 0; j < 4; j++)
        printf("%d \t", matr[i][j]);

  printf("\n");
 }
```

Task 32

Create a program that contains an array that has this structure column contains a year
1. column contains the population of the world Put there some 5 rows.
Print it.
Search the population of some year.

from https://en.wikipedia.org/wiki/World_population
we get this info

1,1804,
2,1927,
3,1960,
4,1974,
5,1987,
6,1999,
7,2011,
8,2022,
9,2037,
10,2057

Solution

```
            int pops[10][2] = {
 1,1804,
 2,1927,
```

```
    3,1960,
    4,1974,
    5,1987,
    6,1999,
    7,2011,
    8,2022,
    9,2037,
    10,2057
};

   int i, j;
   for (i = 0; i < 10; i++)
   {
    for (j = 0; j < 2; j++)
           printf("%d \t", pops[i][j]);

     printf("\n");
    }

   int year = 2011;

   for (i = 0; i < 10; i++)
   {
    if (pops[i][1] == year)
       {
           printf("%d \n", pops[i][0]);
      break;}
   }
```

Task 33

Create a program that contains an array that has this structure
row 1 contains the population of some country
row 2 contains the area of that country
row 3 is empty
Calculate the population density to 3. row.

Some info about orthern countries, we take finland and Sweden with now

Country	Inhabitants	Area
Denmark	5,806,014	42,933
Finland	5,520,535	338,424
Norway	5,323,933	385,203
Sweden	10,313,447	450,295

Solution

```
float info[3][2];

// Finland
info[0][0] = 5806014;
info[1][0] = 338000;

 // Sweden
info[0][1] = 10313447;
info[1][1] = 450000;

// pop.densities (now manually)
info[2][0] = info[0][0]/info[1][0];
info[2][1] = info[0][1]/info[1][1];

printf("%f \n",  info[2][0]);
printf("%f \n",  info[2][1]);
```

Task 34

Create a program that contains an array that has this structure and values.

1,5,6,6,7,7,
2,4,6,8,8,8,
3,5,5,8,6,8,
4,9,6,8,5,8,
5,7,6,7,8,10

1. column is the order of measurement set Columns 2-6 contain measurement values
Search for the biggest average of those measurement sets

Solution

```
int measures[5][6] =
{1,5,6,6,7,7,
2,4,6,8,8,8,
3,5,5,8,6,8,
4,9,6,8,5,8,
5,7,6,7,8,10};

int sums[] = {0,0,0,0,0};

int i, j;
for (i = 0; i < 5; i++)
 for (j = 1; j < 6; j++)
   sums[i] = sums[i] + measures[i][j];
```

```
    for (i = 0; i < 5; i++)
      printf("%d \n", sums[i]);

    float avers[5];
    for (i = 0; i < 5; i++)
      avers[i] = sums[i]/5.0;

    for (i = 0; i < 5; i++)
      printf("%f \n", avers[i]);

    float max = avers[0];
    for (i = 0; i < 5; i++)
      if (avers[i] > max)
         max = avers[i];

 printf("Max. average is %f \n", max);
```

Task 35

Your array has these values
1, 2, 5, 8, 4, 2, 3, 22, 33, 11, 0, 5

Write a program that tells how many values are bigger than 10.

Solution

```
    int vals[] =  {1, 2, 5, 8, 4, 2, 3, 22, 33, 11, 0, 5};

     int over10 = 0;
     int i;
    for (i = 0; i < 12; i++)
          if (vals[i] > 10)
               over10++;

        printf("Over 10 are %d values \n", over10);
```

Task 36

Create a program that contains an array that contains 8 measurements.
Calculate the standard deviation. Compare the result to Excel result.

Solution

```
    float meas[] = {1.1, 1.5, 1.7, 2, 2.6, 2.4, 3.5, 4.5};
     float aver, sum;
     int i;

     for (i = 0; i < 8; i++)
```

```
    sum += meas[i];

aver = sum/8;

float temp_value = 0;

for (i = 0; i < 8; i++)
    temp_value = temp_value + (meas[i] - aver) * (meas[i] -
                 aver);

 float std = sqrt(temp_value/7);

 printf("std is %f \n", std);
```

32

SET 5: Functions

Task 37

Create a function that:
Calculates the sum of 2 integers and prints out the result.

Solution

```
void print_values(int a, int b)
{
    printf("%d %d \n", a, b);
}
```

Task 38

Create a function that:
Returns the sum of 2 integers.

Solution

```
int calc_sum(int a, int b)
{
    return (a + b);
}
```

Task 39

Create a function that:
Returns the average of 2 integers

Solution

```
float calc_aver(int a, int b)
{
    return ((a + b)/2.0);
}
```

Task 40

Create a function that:
Returns the average of 4 floating point values.

Solution

```
float calc_aver_of_4(float a, float b, float c, float d)
{
    return ((a + b + c + d)/4);
}
```

Task 41

Create a function that:
Returns the factorial.

Solution

```
int fact(int n)
{
    int f = 1;
    int i;

    if (n == 0)
      f = 1;
    else
    for (i = 1; i <= 7; i++)
    {
     f = f * i;
    }

    return f;
}
```

Task 42

Create a function that:
Returns bigger of 2 integers.

Solution

```
int bigger(int a, int b)
{
    if (a > b)
      return a;
    else
      return b;
}
```

Task 43

Create a function that:
Returns the biggest of 3 integers.

Solution

```
int biggest_of_3(int a, int b, int c)
```

```
{
    int max;
    if (a > b && a > c)
      max = a;
    else if (b > a && b > c)
      max = b;
    else
      max = c;

    return max;
}
```

Note: there are more solutions than one...

Task 44

Create a function that:
Converts inches to centimeters.

Solution

```
float inches_to_cm(float inches)
{
    return 2.54 * inches;
}
```

Task 45

Create a function that:
Returns the BMI.

Solution

```
float bmi(float w_kg, float h_cm)
{
   float bmi =   w_kg/(h_cm/100*h_cm/100);
   return bmi;
}
```

Task 46

Create a function that:
Function returns the biggest of 5 integers.

Solution

```
int biggest_5(int a, int b, int c, int d, int e)
{
    int maks = a;
    if (b > maks)
      maks = b;
```

```
    if (c > maks)
      maks = c;
    if (d > maks)
      maks = d;
    if (e > maks)
      maks = e;

    return maks;
}
```

Task 47

Program with functions calculates amount of combinations.

```
int fact(int p)
{
    int kert = 1;
    int i;
    for (i = 1; i <= p; i++)
    {
     kert = kert * i;
    }

    return kert;
}

int kombin(int n, int k)
{
    int tulos = fact(n)/(fact(n-k) * fact(k));
    return tulos;
}
```

Task 48

Function prints out a lotto row.

Solution

```
void lotto()
{
int nros[]  = {0,0,0,0,0,0,0};

   int i;
   for (i = 0; i < 7; i++)
   {

      int existed_already = 0;
```

```
        int newnr = rand() % 40 + 1;
        int j;
        for (j = 0; j <= i; j++)
        {
             if (nros[j] == newnr)
             {
                   existed_already = 1;
                   break;
           }
         }

         if (existed_already == 1)
           i--;
         else
           nros[i] = newnr;

    }
     for (i = 0; i < 7; i++)
       printf("%d ", nros[i]);
    return 0;
}

}
```

Task 49

Program with functions calculates the standard deviation.

Solution

```
float std()
{
float meas[] = {1.1, 1.5, 1.7, 2, 2.6, 2.4, 3.5, 4.5};
     float aver, sum;
     int i;

     for (i = 0; i < 8; i++)
        sum += meas[i];

    aver = sum/8;

    float temp_value = 0;

    for (i = 0; i < 8; i++)
        temp_value = temp_value + (meas[i] - aver) * (meas[i] -
aver);

     float std = sqrt(temp_value/7);
```

```
      printf("std is %f \n", std);
   */


}
```

Task 50

Program with functions calculates the sum on an array.

Solution

```
int sum_of_array(int array[], int n)
{

     int sum = 0;
     int i;
     for (i = 0; i < n; i++)
     {
      sum += array[i];
     }
     return sum;
}
```

Task 51

A character is passed to a function: funtion returns True if character is a vowel, otherwise False (0).
(Five of the 26 alphabet letters are vowels: A, E, I, O, and U.)

Solution

```
int is_vowel(char c)
{
        int result = 0;
        switch (c)
        {
                case 'a': result = 1; break;
                case 'e': result = 1; break;
                case 'i': result = 1; break;
                case 'o': result = 1; break;
                case 'u': result = 1; break;
   }

        return result;
}
```

Task 52

A whole number and an array (size is 5, contains integers) are passed to

a function that checks how many times passed value exists in that passed array and returns the amount.

Solution

```
int amount_of_val(int vals[], int n, int x)
{
     int amount = 0;
     int i;
     for (i = 0; i < n; i++)
     {
      if (vals[i] == x)
      amount++;
     }

     return amount;

}
```

Task 53

Your program defines and fills an array of 10 integers with random numbers that are between 1-5.
That array is passed to a method that counts the amounts of different values and prints then out.

Solution

```
int amounts_of_diff_vals(int vals[])
{
     int difs = 0;
     int i, j;
     int sample[] = {0,0,0,0,0};
     for (i = 0; i < 10; i++)
       switch (vals[i])
       {
            case 1: sample[0]++; break;
            case 2: sample[1]++; break;
            case 3: sample[2]++; break;
            case 4: sample[3]++; break;
            case 5: sample[4]++; break;

       }

     for (i = 0; i < 5; i++)
       printf("%d: %d \n", i+1, sample[i]);

}
```

```
Main:
 int array[10];
    int i;
    for (i = 0; i < 10; i++)
    {
     array[i]  = rand() % 5 + 1;
    }

    amounts_of_diff_vals(array);
```

Task 54

Duration and frequency are passed to a function that plays then that sound (windows.h neeeded).

Solution

```
void play_this(int freq, int dur)
{
    Beep(freq, dur);

}

Main:
    play_this(200,500);
    play_this(400,500);
    play_this(600,500);
    play_this(800,500);
```

Note:
#include <windows.h>

Task 55

Function converts the text to morse code characters.

Solution

```
void morse_this(char message[])
{
    int p = 0;
    // length
    for (p = 0; message[p] != NULL; p++);
    int i;
    for (i = 0; i < p; i++)
    {
     switch(message[i])
     {
          case 'O': printf("--- "); break;
```

```
            case 'S': printf("... "); break;
            case ' ': printf("  "); break;
    }
}

}
```

Task 56

Function returns the range value of an array that has 5 whole numbers and that is passed to the function. Range means: max – min.

Solution

```
float range(float vals[], int n)
{
    float max = vals[0];
    float min = vals[0];

    int i;

    for (i = 0; i < n; i++)
    {
     if (vals[i] < min)
        min = vals[i];
     if (vals[i] > max)
        max = vals[i];
    }

    return (max - min);
}
int main()
{
  float values[] = {6.0, 2.5, 2.6, 3.0, 5.0};

  printf("range is %f \n", range(values, 5));

  return 0;
}
```

SET 6: Strings

Strings & dynamic memory allocation

Task 57

User is asked to give the amount of values.
Then a new array is created.
It is filled with randon numbers.
10 first are then printed

Solution

```
void fill(int arr[],int n)
{
     int i;
     for (i = 0; i < n; i++)
      arr[i] = rand() % 100;

}

void print(int arr[],int n)
{
     int i;
     for (i = 0; i < n; i++)
      printf("%d \n", arr[i]);

}


int main()
{
  int amount;
  int * values;
  printf("How big array is to be created \n");
  scanf("%d", &amount);
  values = calloc(amount, 4);
  fill(values, amount);
  print(values, 10);
}
```

Task 58

Function checks if the post code includes exactly 5 numbers

Solution

```
int check_post_code(char text[],int n)
```

```
{
     int i;
     int res = 1;
     if (n != 5)
        { res = 0; return res;}
     else
     {
      for (i = 0; i < n; i++)
        if (text[i] < '0'  && text[i] > '9')
        {
            res = 0;
            break;
        }
     }

     return res;
}

int main()
{

   char * postcode = "2233";
     int p = 0;
     // length
     for (p = 0; postcode[p] != NULL; p++);
    printf("%d \n", check_post_code(postcode, p));

}
```

Task 59

Program checks if an email-address contains '@' character.

Solution

```
char * email = "ducks@ducks.com";

int on = -1;

   int p = strlen(email);

     int i;
     for (i = 0; i < p; i++)
     {
      if (email[i]  == '@')
      {
            on = i;
            break;
```

```
  }
 }

 if (on == -1)
    printf("EI oo \n");
 else
   printf("O. \n");
```

Task 60

Program prints out the country code (top level domain name) of an url.

Solution

```
 char * url = "www.vossilos.com";

 int lastdot;

 int p = strlen(url);

 int i;
 for (i = 0; i < p; i++)
 {
  if (url[i]  == '.')
  {
        lastdot = i;
  }
 }

 for (i = lastdot; i < p; i++)
 {
  printf("%c", url[i]);

 }
```

Task 61

Program prints out the protocol of an url.

Solution

```
 char * url = "https://www.vossilos.com";

 int colonplace;

 int p = strlen(url);

 int i;
 for (i = 0; i < p; i++)
 {
  if (url[i]  == ':')
```

```
  {
        colonplace = i;
        break;
  }
 }

for (i = 0; i < colonplace; i++)
{
 printf("%c", url[i]);

}
```

Task 62

Program tells if a string is a palindrome.

Solution

```
 char * word = "rotator";

    int size = strlen(word);

    char newword[size];

    int s = 0;
    int j;
    for (j = size - 1; j >= 0; j--)
    {
     newword[s] = word[j];
     s++;
    }

    printf("%s \n", word);
    printf("%s \n", newword);

    if (strcmp(word, newword) == 0)
    printf("Is a pal... \n");
    else
    printf("Is not a pal... \n");
```

Task 63

String variable contains 5 measures separated by commas. Your program calculates the average of those values. (E.g. "2, 3.5, 1, 5.8, 10") is given.)

Solution

```
char * row = "2, 3.5, 1, 5.8, 10";
int point_places[4];
int size = strlen(row);
int i;
int j = 0;
for (i = 0; i < size; i++)
{
 if (row[i] == ',')
 {
   point_places[j] = i;
   j++;
}
}

for (j = 0; j < 4; j++)
  printf("%d ", point_places[j]);

   printf("\n");

char * val1 = malloc(point_places[0] + 1);
char * val2 = malloc(point_places[1] - point_places[0] + 1);
char * val3 = malloc(point_places[2] - point_places[1] + 1);
char * val4 = malloc(point_places[3] - point_places[2] + 1);
char * val5 = malloc(size  - point_places[3] + 1);

for (j = 0; j < point_places[0]; j++)
  val1[j] = row[j];

val1[j] ='\0';

printf("%s ", val1);

   printf("\n");

int s = 0;
for (j = point_places[0] + 1; j < point_places[1]; j++)
  val2[s++] = row[j];

val2[s] ='\0';
```

```
printf("%s ", val2);

   printf("\n");

 s = 0;
for (j = point_places[1] + 1; j < point_places[2]; j++)
  val3[s++] = row[j];

val3[s] ='\0';

printf("%s ", val3);

   printf("\n");

  s = 0;
for (j = point_places[2] + 1; j < point_places[3]; j++)
  val4[s++] = row[j];

val4[s] ='\0';

printf("%s ", val4);

   printf("\n");

 s = 0;
for (j = point_places[3] + 1; j < size; j++)
  val5[s++] = row[j];

val5[s] ='\0';

printf("%s ", val5);

   printf("\n");

float a = atof(val1);
float b = atof(val2);
float c = atof(val3);
float d = atof(val4);
float e = atof(val5);

float avg = (a + b + c + d + e)/5;

printf("%f", avg);
```

Task 64

Read a NMEA sentence and print latitude and longitude.
Info here

NMEA-0183 message: GGA

Related Topics

- NMEA-0183 messages: Overview

Time, position, and fix related data

An example of the GBS message string is:

```
$GPGGA,172814.0,3723.46587704,N,12202.26957864,W,2,6,1.2,18.893,M,-25.669,M,2.0
0031*4F
```

NOTE – The data string exceeds the NMEA standard length.

GGA message fields

Field	Meaning
0	Message ID $GPGGA
1	UTC of position fix
2	Latitude
3	Direction of latitude: N: North S: South
4	Longitude

Solution

```
#include <stdio.h>
#include <stdlib.h>

int main()
{
  char nmea_sentence[] =
"$GPGGA,172814.0,3723.46587704,N,12202.26957864,W,2,6,1.2,18.893,M,-
25.669,M,2.0,0031*4F";

  int point1, point2, point3, point4

  int p = 0;

  int i;
  for (i = 0; i < strlen(nmea_sentence); i++)
  {
    if (nmea_sentence[i] == ',')
    {
    p++;
```

```
      if (p == 2)
        point1 = i;

      if (p == 3)
        point2  = i;

      if (p == 4)
        point3 = i;

      if (p == 5)
         point4 = i;
  }
    }

    printf("Latitude is \n");
    for (i = point1 + 1; i < point2; i++)
      printf("%c", nmea_sentence[i] );

    printf("\nLongitude is \n");
    for (i = point3+1; i < point4; i++)
      printf("%c", nmea_sentence[i] );

    return 0;
   }
```

Task 65

Check that given country code has exactly 1..3 numbers.
Country codes are listed here...
https://www.iban.com/country-codes

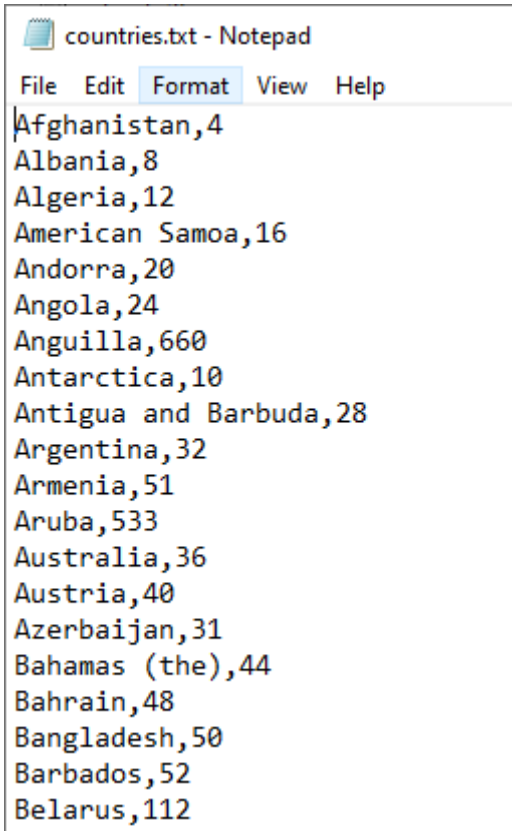
Solution

```
#include <stdio.h>

int main()
{
  char code[5];

  printf("Give the country code: \n");

  scanf("%s", code);

 // printf("%s %d \n", code, strlen(code));

  int accepted = 0;
```

```
  if (strlen(code) == 3)
  {
    if (code[0] >= '0' && code[0] <= '9'  &&
     code[1] >= '0' && code[1] <= '9' &&
     code[2] >= '0' && code[2] <= '9')
     accepted = 1;
   else
     accepted = 0;
  }

  if (accepted == 0)
    printf("wrong country code \n");
  else
   printf("correct country code \n");

  return 0;
 }
```

Task 66

Information of countries are to be taken to a textfile.
Country info is taken to a text file, example here.



Read the file and print it.

Solution

```
FILE *filepointer;
 char row[256];

 char name[] = "countries.txt";

 filepointer = fopen(name,"r");

 if (filepointer == NULL)
 {
  printf("Can not open the file \n");
  return;
 }

 int i, j;

 while(fgets(row, 100, filepointer))
 {
 printf("%s \n", row);
 }
```

Task 67

In Finland, the Personal Identity Code (Finnish: henkilötunnus (HETU), Swedish: personbeteckning) also known as Personal Identification Number consists of eleven characters of the form DDMMYYCZZZQ, where DDMMYY is the date of birth, C the century sign, ZZZ the individual number and Q the control character (checksum). Check given code.

Solution

```
char * sotu = "040363-011X";

    char nrs[9];
    nrs[0] = sotu[0]; nrs[1] = sotu[1];nrs[2] = sotu[2];
    nrs[3] = sotu[3];nrs[4] = sotu[4];nrs[5] = sotu[5];
    nrs[6] = sotu[7];nrs[7] = sotu[8];nrs[8] = sotu[9];

    printf("%s \n", nrs);
    int as_value = atol(nrs);
    printf("%d \n", as_value);

    int div = as_value % 31;

    char * right_chars = "0123456789ABCDEFHJKLMNPRSTUVWXY";
    printf("%c \n", right_chars[div]);
```

```
    if (right_chars[div] == sotu[11])
        printf("YEAH");
    else
     printf("NONO");
```

Task 68

Create a Finnish Italian dictionary. Take words from some Internet place and add them to an array.

Solution

```
 FILE *file;
 char italian_word[256];
 char english_word[256];

 char * search_this = "parola";

 char nimi[] = "words.txt";

 file = fopen(nimi,"r");


 if (file == NULL)
 {
  printf("Can not open \n");
  return;
 }

 char word[200];

char * rows[2000];
int i = 0;
while (!feof(file))
{
     fgets(word,200, file);
     printf("%s \n",word);
     strcpy(rows[i], word);
     i++;
}


for (i = 0; i < 1000; i++)
  printf("%s \n", rows[i]);
```

```
/*


while (!feof(file))
{
  fgets(italian_word, 200, file);
 // printf("%s \n", italian_word);

 if (strcmp("italian_word,", search_this) == 0)
 {
    fgets(english_word, 200, file);
    printf("%s", english_word);
    printf("%s", italian_word);
     break;
 }

}
```

SET 7: Struct

Task 69
Create a struct that models a Dot.
Create the 2 dots in your program.

Add then 20 dots to an array (use random numbers)
Print dots

Calculate the distance of first and last dots.

Solution

```
struct Dot
{
    int x;
    int y;
};

typedef struct Dot Point;


int main () {

    Point p1;
    p1.x = 5;
    p1.y = 6;

    Point p2;
    p2.x = 2;
    p2.y = 9;

    Point points[20];
    int i;
    for (i = 0; i < 20; i++)
    {
     int xx = rand() % 50;
     int yy = rand() % 50;
     points[i].x = xx;
     points[i].y = yy;
    }

    for (i = 0; i < 20; i++)
    {
     printf("(%d,%d) \n", points[i].x, points[i].y);
    }
```

```
int xx1, yy1, xx2, yy2;
xx1 = points[0].x;
yy1 = points[0].y;

xx2 = points[19].x;
yy2 = points[19].y;

double et = sqrt((xx1 - xx2)*(xx1-xx2) + (yy1-yy2)*(yy1-yy2));

printf("%f \n", et);
```

SET 8: Header files

Task 70

Program calculates the hypotenuse of an triangle when other sides are given.

Solution

```
float a, b, c;
a = 3;
b = 4;
c = sqrt(a*a + b*b);

printf("c = %f \n", c);
```

Task 71

Program rounds a double value to a value that has 2 numbers in its fractional part.

Solution

```
float x = 22.4567;
    float y = round(100*x+0.5)/100;
    printf("y = %.2f \n", y);
```

Task 72

Program tells how much time does it take to sort an array of 100000 elements.
Compare sorting times to time got from c:s own sort() function.

Solution

```
printf("elapsed time now is %d secs\n", time(NULL));

  int size = 150000;

  int vals[size];

  int i;
  for (i = 0; i < size; i++)
  {
    vals[i] = rand();
  }

  /*
  for (i = 0; i < 20; i++)
  {
    printf("%d \n", vals[i]);
  }
  */
```

```
  int time1 = time(NULL);
  // selection sort
  int m, n, temp;
  for (m = 0; m < size; m++)
    for (n = m + 1; n < size; n++)
    {
     if (vals[n] < vals[m])
     {   // swap
           temp = vals[n];
           vals[n] = vals[m];
           vals[m] = temp;
        }
     }

  int time2 = time(NULL);

  printf("I took %d secs \n", (time2 - time1));

/*
  printf("\n Sorted: \n");
  for (i = 0; i < 20; i++)
  {
    printf("%d \n", vals[i]);
  }
*/

// QSORT example

int values[] = { 88, 56, 100, 2, 25 };

int cmpfunc (const void * a, const void * b) {
   return ( *(int*)a - *(int*)b );
}

int main () {
   int n;

   printf("Before sorting the list is: \n");
   for( n = 0 ; n < 5; n++ ) {
      printf("%d ", values[n]);
   }

   qsort(values, 5, sizeof(int), cmpfunc);

   printf("\nAfter sorting the list is: \n");
   for( n = 0 ; n < 5; n++ ) {
      printf("%d ", values[n]);
   }
```

Task 73

Calculate the square root of some value using numeric method and compare the result to the value got with sqrt() function.

Solution

```
     float a = 5;
     float c = 0.3;
     while (1)
     {
      if ((c*c - a) > -0.1 && (c*c - a) < 0.1 )
            break;
      printf("c = %f and diff is %f \n", c, c*c - a);
      c = c * 1.1;
     }
    printf("c = %f \n", c);
```

Task 74

Calculate approximations of Nepers's value, pi and cos(0.9) and compare them t values of got from math.h functions.

Solution

```
int fact(int n)
{
     int kert = 1;
     int i;
     for (i = 1; i <= n; i++)
     {
      kert = kert * i;
     }

     return kert;
}

int main()
{
     int j;
     float e = 1;
     for (j = 1; j < 10; j++)
     {
     e  = e + 1.0/fact(j);
     }

     printf("%f \n", e);

     printf("%f \n", M_E);
```

Task 75

Program throws dice 100 times and tells amounts of different values (1, 2, 3, 4, 5, and 6).

Solution

```
srand(time(NULL));


int n1 = 0;  int n2 = 0;  int n3 = 0;
int n4 = 0;  int n5 = 0;  int n6 = 0;

int i;
for (i = 0; i < 10000; i++)
{
    int noppis = rand() % 6 + 1;

    switch (noppis)
    {
        case 1: n1++; break;
        case 2: n2++; break;
        case 3: n3++; break;
        case 4: n4++; break;
        case 5: n5++; break;
        case 6: n6++; break;
    }
}

 printf("1: %d \n", n1);
 printf("2: %d \n", n2);
 printf("3: %d \n", n3);
 printf("4: %d \n", n4);
 printf("5: %d \n", n5);
 printf("6: %d \n", n6);
```

Task 76

Create this array

Population, thousands

	1900	1950	2000	2018	2019
Total	2 656	4 030	5 181	5 518	5 525
Males	1 311	1 926	2 529	2 723	2 728
Females	1 345	2 104	2 652	2 795	2 797

Print it in good format

Give a year and use your array: then your program tells how many males and females Finland had in that year.

Table is here also as text if needed:

	1900	1950	2000	2018	2019
Total	2 656	4 030	5 181	5 518	5 525
Males	1 311	1 926	2 529	2 723	2 728
Females		1 345	2 104	2 652	2 795	2 797

Solution

```
char * stat[4][6] =
{"   ", "1900","1950","2000","2018","2019",
"Total","2 656","4 030","5 181","5 518","5 525",
"Males","1 311","1 926","2 529","2 723","2 728",
"Females","1 345","2 104","2 652","2 795","2 797"};


int r, s;

for (r = 0; r < 4; r++)
{

    for(s=0; s < 6; s++)
       printf("%s ", stat[r][s]);

    printf("\n");
}

   int year = 2000;
   printf("\n");

    for(s=0; s < 6; s++)
    {
```

```
      if (year  == atoi(stat[0][s]))
      {
            printf("males: %s,  females:  %s ", stat[2][s], stat[3]
[s]);
            break;
        }
    }
```

Task 77

Program tells how many big letters (capital letters) does a string contain.

Text is here:
The EEA includes EU countries and also Iceland, Liechtenstein and Norway.
It allows them to be part of the EU's single market.
Switzerland is not an EU or EEA member but is part of the single market.
This means Swiss nationals have the same rights to live and work in the UK as
other EEA nationals.

Solution

```
char text[] =
"The EEA includes EU countries and also Iceland, Liechtenstein and
Norway. It allows them to be part of the EU\'s single
market.Switzerland is not an EU or EEA member but is part of the
single market.";


int bigs = 0;
int i;
for (i = 0; i < strlen(text); i++)
{
    if (text[i] >= 65 && text[i] <= 90)
    bigs++;
}
printf("%d ", bigs);
```

Task 78

A stone is dropped down from the top of Pisa tower.
What is the final speed of the stone and how much time does the fall take?

Solution

```
    //  v = s/t        a = v/t      a = g
    //  g = v/t     | *t => gt = v  |:g  => t = v/g  => t = s/t */g
| *t  => t^2 = s/g
    // => gt^2 = s =>  t^2 = s/g
    //  t = sqrt(s/g)
    //  g = 9.81 m/sek^2  => t = sqrt(57/9.81)  => 2,3 sek
```

```
//  v = 57/2,3  => 24,5 m/sek  => 88 km/h

        float s = 57; // m
float g = 9.81;
float t = sqrt(s/g);

printf("%f \n", t);

float v = s/t;

printf("%f \n", v);

v = (s/1000)/(t/3600);

printf("%f \n", v);
```

Task 79

Create an array of given animal data.

Then print it.

Here is part of the list that is taken from Internet:

"African Grey Parrot,50",

"Alligator,68",

"Amazon Parrot,80",

"American Alligator,56",

"American Box Turtle,123",

"American Newt,3",

… continues

Solution

```
char * animals[] = {
"African Grey Parrot,50",
"Alligator,68",
….

// amount of items
printf("%f \n", sizeof(animals)/8.0);

int i;

for (i = 0; i < 154; i++)
   printf("%s \n", animals[i]);
```

Task 80

Standard deviation

Solution

```
 int vals[100];
     int k;
     for (k = 0; k < 100; k++)
       vals[k] = rand();

     int sum  = 0;
     for (k = 0; k < 100; k++)
         sum = sum + vals[k];

     double ka = sum/100.0;

     double std = 0;
     for (k = 0; k < 100; k++)
         std = std + (vals[k] - ka)*(vals[k] - ka);

     std = std/100;
     std = sqrt(std);

     printf("%f \n", std);
```

Task 81

Students's grades (Swedish, Math) are added to arrays. Calculate the correlation.

Solution

```
double sve[] = {1.5, 2.5, 1, 5.0, 3.5};
     double math[] = {4.5, 1.5, 1.5, 3.0, 2.5};

     double s1 = 0;
     int k;
     for (k = 0; k < 5; k++)
         s1 = s1 + sve[k];

     double s2 = 0;
     for (k = 0; k < 5; k++)
         s2 = s2 + math[k];

     double ka1 = s1/5;
     double ka2 = s2/5;

     double summa1 = 0;
     for (k = 0; k < 5; k++)
         summa1 = summa1 + (sve[k] - ka1)*(math[k] - ka2);
```

```
double summa2 = 0;
for (k = 0; k < 5; k++)
    summa2 = summa2 + (sve[k] - ka1)*(sve[k] - ka1);


double summa3 = 0;
for (k = 0; k < 5; k++)
    summa3 = summa3 + (math[k] - ka2)*(math[k] - ka2);


double korre = summa1/sqrt(summa2*summa3);

printf("correlation is %f \n", korre);
```

SET 9: Bitwise operators

Task 82
Create a program that uses all bit operators that are shown in the table below.

So, create 2 integer variables. Assign values and test AND, OR and XOR.
Then try shift operators with one variable.
Print also results.

Here are bitwise operators

Operator	Meaning
&	AND
|	OR
<<	Left shift
>>	Right shift
~	One's complement
^	XOR

Solution

```
int a = 199; // 1100 0111
int b = 222; // 1101 1110
int c;

// AND  &
/*
11000111
11011110
11000110    => 198
*/
c = a & b;
printf("a & b is %d \n", c);

// OR  |
/*
11000111
11011110
11011111   => 223
*/
c = a | b;
printf("a | b is %d \n", c);

// XOR ^
/*
11000111
```

```
11011110
00011001   => 25
*/
c = a ^ b;
printf("a ^ b is %d \n", c);

// shift value a 2 times to the left:   a << 2
/*
11000111   << 2
1100011100  => 796
*/
 c = a << 2;
 printf("a << 2 is %d \n", c);


// shiftvalue of variable a  once to the right   a >> 1
/*
11000111   >> 1
01100011   =>  99
*/
 c = a >> 1;
 printf("a >> 1 is %d \n", c);
```

Task 83

Check the state of given bit in a bit queue

Tips: Right shift the original bit queue until the bit that has to be inverted is the first bit. Then take bitwise AND between 1 and shifted bit queue. You get the state of the wanted bit.

Solution

```
We have value 155 in a variable. As bits it is 10011011.
We want to know the 3. bit's state.  (LSB s now position 0).
So we shift 155 3 times to the right and get  00010011.
Then we take AND between that new bit queue and value 1 and we get
0000 0001
that tells that state is 1.

int a = 155;
n = 3;
int state = (a >> n) & 1;
printf("state is %d \n", state);
```

Task 84

Invert the given bit in a bit queue.

Tips: Create a bit mask that has bits 0 and where value 1 has the same position than the bit

that is to b inverted. Then take Xor between the mask and the original bit queue. The result is a new bit queue where wanted bit is inverted.…

Solution

```
int a = 155;
n = 4;
int mask = 1 << (n - 1);
a = a ^ mask;
printf("a is now %d \n", a);
```

SET 11: Time

Task 85

Print message ”Life is wonderful” a) 5000 times and b) 10000 times. How long time do it take? Use time.h and time() function.

```
Solution (a)
#include <stdio.h>
#include <time.h>

int main()
{
    int i;
    int n = 5000;

    int time1 = time(0);
    for (i = 0; i < n; i++)
    {
     printf("Life is wonderful ...");
    }

    int time2 = time(0);

    printf("It took %d secs \n", (time2-time1));

    return 0;

}
```

Task86

Create an array that contains 1 million random numbers. Sort that array using qsort that is declared in stdlib.h

```
Solution
#include <stdio.h>
#include <time.h>
#include <stdlib.h>

int compare(const void* a, const void* b) {
   return (*(int*)a - *(int*)b);
}

int main()
{
    int n = 1000000;
    int * vals = calloc(n, sizeof(int));
```

```
    int i;

    for (i = 0; i < n; i++)
    {
     vals[i] = rand();
    }

    for (i = 0; i < 10; i++)
    {
          printf("%d \n", vals[i]);
    }

    qsort(vals, n, sizeof(int), compare);

    for (i = 0; i < n; i += 100000)
    {
          printf("%d \n", vals[i]);
    }

    return 0;

}
```

Task 87

Create an array that contains 1 million random numbers. Sort that array using selection sort

Solution

```
#include <stdio.h>
#include <time.h>
#include <stdlib.h>

void selsort(int v[], int n)
{
    int i, j;
    for (i = 0; i < n; i++)
       for (j = i; j < n; j++)
          if (v[i] > v[j])
          {
           int temp = v[i];
           v[i] = v[j];
           v[j] = temp;
          }
}

int main()
{
    int n = 1000000;
```

```
    int * vals = calloc(n, sizeof(int));
    int i;

    for (i = 0; i < n; i++)
    {
     vals[i] = rand();
    }

    for (i = 0; i < 10; i++)
    {
          printf("%d \n", vals[i]);
    }

    selsort(vals, n);

    for (i = 0; i < 10; i++)
    {
          printf("%d \n", vals[i]);
    }


    return 0;
}
```

Task 88

Use solutions of tasks 83 and 85 and take execution times. Can you tell a reason why another sorting method is so slow?

Solution

```
Main code for timetaking;

int time1 = time(0);
// here sorting
int time2 = time(0);

printf("It took %d secs \n", (time2-time1));
```

Selection sort is very very slow: there are a lot of comparisons and swaps. It is $O(n^2)$ algorithm.

SET 13: Miscellanious

Task 89

Program checks if given email address contains @. Use own code and then library function.

Solution

a)

```
          char * email = "nicke.nacke@ducks.com";
    int pituus = strlen(email);
    int on = 0;
    int i;
    for (i = 0; i < pituus; i++)
    {
       if (email[i]   == '@')
       {
             on = 1;
             break;
          }
    }

    if (on == 1)
       printf("OK \n");
    else
      printf("EI OK \n");
```

b)

```
    int paikka = strchr(email, '@');
    printf("paikka on %d \n", paikka);
```

Task 90

A Finnish lottogame contains values 1 – 40. From those values are 7 values chosen randomly to form a lottorow.

Solution

```
    srand(time(NULL));
    int nrs[] = {0,0,0,0,0,0,0};
    int exists  = 0;
    int i, j;
    for (i = 0; i < 7; i++)
    {
     exists  = 0;
     int x = rand() % 40 + 1;
     nrs[i] = x;
     for (j = 0; j < i; j++)
           if (nrs[j] == x)
          {
```

```
            exists = 1;
          break;
        }

  if (exists == 1)
            i--;
     else
        nrs[i] = x;
  }


 for (i = 0; i < 7; i++)
 {
     printf("%d, ",nrs[i]);
 }
```

SET 14: Linked lists

Task 91

Create a linked list that contains these values: 10, 33, 7777. Then print the list.

Solution

```
#include <stdio.h>

    struct ITEM
    {
        int value;
        struct Item * next;
    };


    // 10, 33, 7777

int main()
{
        typedef struct ITEM Item;

        Item * start = malloc(sizeof(Item));

        start->value = 10;
        printf("Value 1 is %d \n", start->value);

        Item * end = start;


        Item * temp = malloc(sizeof(Item));
        temp->value = 33;
        printf("Value  is %d \n", temp->value);

        end->next = temp;
        end = temp;

        temp = malloc(sizeof(Item));
        temp->value = 7777;
         printf("Value  is %d \n", temp->value);

        end->next = temp;
        end = temp;

        end->next = NULL;

        // print all
        for (temp = start; temp != NULL; temp = temp->next)
```

```
            printf("Value is %d \n", temp->value);


    return 0;
}
```

Task 92

Create a doubly linked list that contains these values 10, 33, 567, -8.
Print the list.

Solution

```
struct ITEM
    {
        int value;
        struct Item * next;
        struct Item * prev;
    };

int main()
{
        typedef struct ITEM Item;

        Item * start = malloc(sizeof(Item));

        start->value = 10;

        start->next = NULL;
        start->prev = NULL;

        Item * end = start;
        Item * temp = malloc(sizeof(Item));
        temp->value = 33;
        end->next = temp;
        temp->prev = end;
        end = temp;

        temp = malloc(sizeof(Item));
        temp->value = 567;
        end->next = temp;
        temp->prev = end;
        end = temp;

        temp = malloc(sizeof(Item));
        temp->value = -8;
        end->next = temp;
        temp->prev = end;
        end = temp;
```

```
        end->next = NULL;

        // print all
        for (temp = start; temp != NULL; temp = temp->next)
           printf("Value is %d \n", temp->value);
     return 0;
}
```

Task 93

Create a Bank Desk SimulationApp.
It has a menu like this:
Check balance
Add money
Withdraw money
Exit

Solution

```
#include <stdio.h>


int main()
{
     int balance = 1000;
     while(1)
     {
     printf("Menu \n");
     printf("1 --> Check balance\n");
     printf("2 --> Take money\n");
     printf("3 --> Add money\n");
     printf("0 --> Exit\n");

     int choice = -1;
     printf("You choice? \n");
     scanf("%d", &choice);


     if (choice == 1)
        printf("Balance is now %d euros \n", balance);
     if (choice == 2)
     {
      int amount;
      printf("How much do you want to withdraw? \n");
         scanf("%d", &amount);
         if (amount <= balance)
         {
            balance -= amount;
            printf("Balance is now %d euros \n", balance);
         }
```

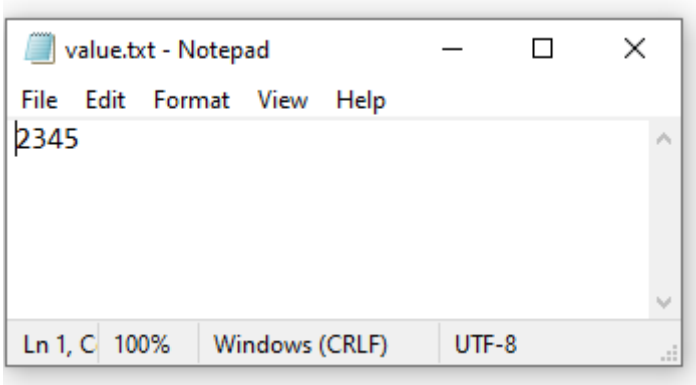
```
        else
        {
           printf("Cannot take so much... Balance is only %d euros
              \n", balance);
        }
     }
           if (choice == 3)
     {
     int amount;
     printf("How much do you want to add? \n");
        scanf("%d", &amount);
        balance += amount;
        printf("Balance is now %d euros \n", balance);
     }

     if (choice == 0)
       break;
    }

    return 0;
}
```

SET 15: Text file handling

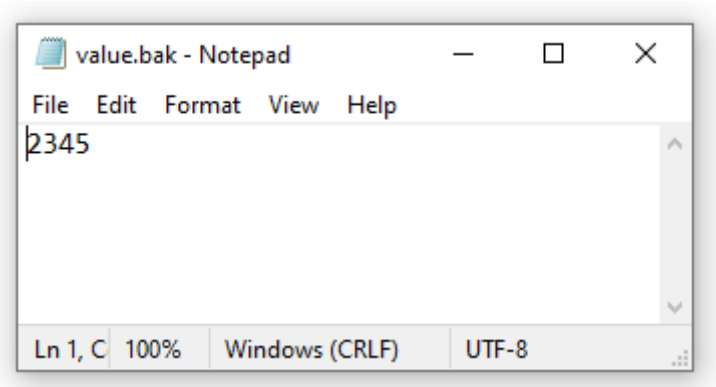
Task 94

Create a program that reads a textfile and saves contents to another textfile.

Here is the textfile:



Here is the new file



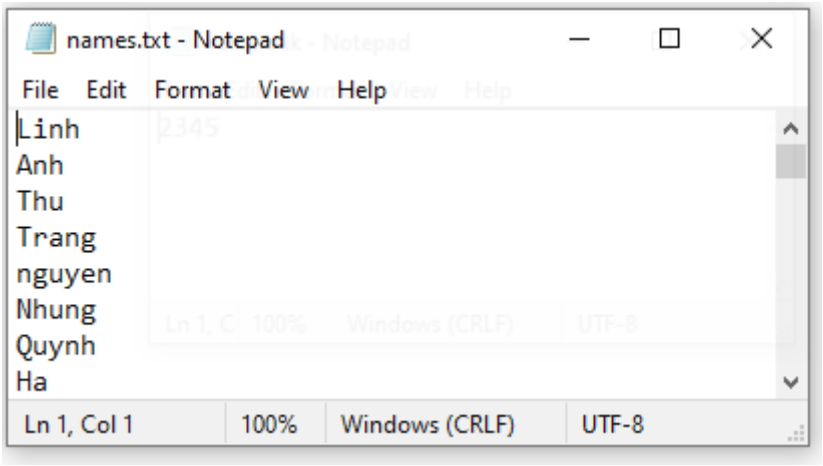
Solution

```
#include <stdio.h>
#include <stdlib.h>

int main()
{
 FILE *filepointer;
 char row[256];

 char name[] = "value.txt";

 filepointer = fopen(name,"r");

 if (filepointer == NULL)
 {
  printf("Can not open the file \n");
  return;
 }

 fgets(row, 200, filepointer);
 fclose(filepointer);

 printf("Print the row\n");
 printf("%s \n", row);



 char *bname = "value.bak";
 filepointer = fopen(bname,"w");

 if (filepointer == NULL)
 {
  printf("Can not open the file \n");
  return;
 }

 fprintf(filepointer,"%s",row);

 fclose(filepointer);

}
```

Task 95

Most popular girl names in Vietnam are here
http://www.studentsoftheworld.info/penpals/stats.php?Pays=VTN

Add names to the textfile named "names.txt":



Print then the list.

Solution

```
#include <stdio.h>
#include <stdlib.h>

int main()
{
 FILE *filepointer;
 char row[256];

 char name[] = "names.txt";

 filepointer = fopen(name,"r");

 if (filepointer == NULL)
 {
  printf("Can not open the file \n");
  return;
 }

while(fgets(row, 100, filepointer))
{
    printf("%s", row);
  }

}
```

Task 96

Add names to an array and print that array

Solution

```
Main code:
 char *  lines[amount_of_rows] ;
 int r = 0;
 while(fgets(row, 100, filepointer))
 {
     lines[r] = malloc(sizeof(row));
     strcpy(lines[r], row);
     r++;
}

for (r = 0; r <  amount_of_rows; r++)
    printf("%s", lines[r]);
```

Task 97

Add names to the linked list and print that list

Solution

```
#include <stdio.h>
#include <stdlib.h>


struct PERSON
{
    char name[20];
    struct PERSON * next;
};


int main()
{
 typedef struct PERSON Person;


 FILE *filepointer;
 char row[256];

 char filename[] = "names.txt";

 filepointer = fopen(filename,"r");

 if (filepointer == NULL)
 {
  printf("Can not open the file \n");
```

```
  return;
 }

Person * first = malloc(sizeof(Person));
Person * last = first;
Person * new_one = first;
new_one->next = NULL;

while(fgets(row, 100, filepointer))
{

    strcpy(new_one->name, row);
    last->next = new_one;
    last = new_one;
    printf("%s ", last->name);
    new_one = malloc(sizeof(Person));
 }

 Person * temp;
 for (temp = first; temp != NULL; temp = temp->next)
    printf("%s ", temp->name);

  return;
 }
```

Task 98

An array contains coordinates of 2 points. You program calculates the distance between thos points.

Example values:
5.5 9
1.7 8

Solution

```
#include <stdio.h>
#include <math.h>

int main()
{

 float points[2][2] =
 {5.5, 9, 1.7, 8};

 float x1 = points[0][0];
 float y1 = points[0][1];

 float x2 = points[1][0];
 float y2 = points[1][1];
```

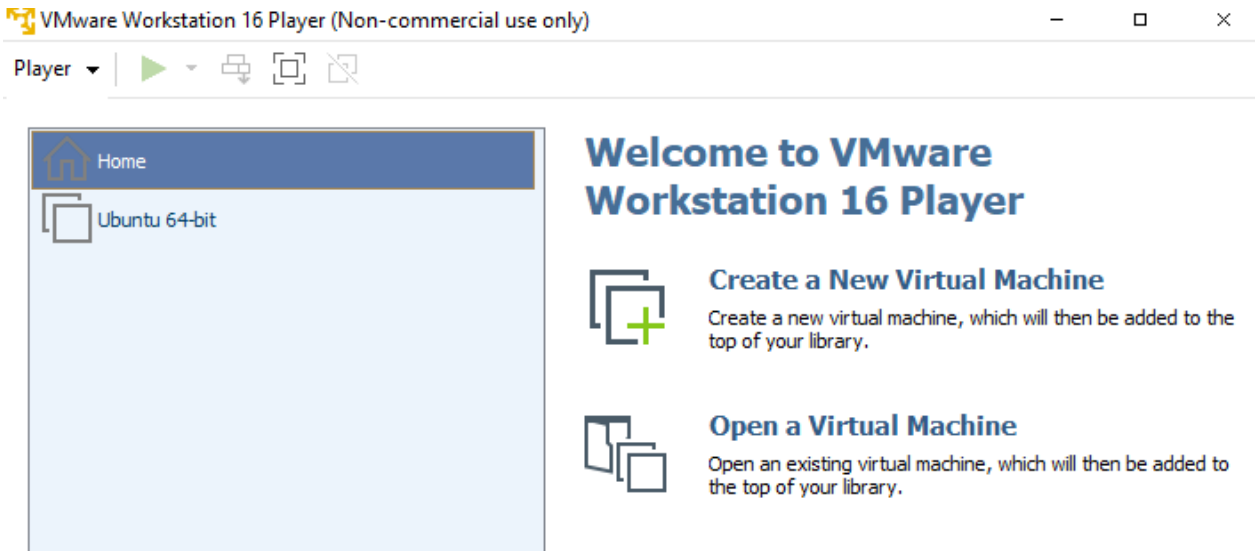
```
 float dist = sqrt((x1 - x2) * (x1 - x2) + (y1 - y2) * (y1 - y2));
 printf("%f", dist);
}
```

SET 16: C coding in Linux

Task 99

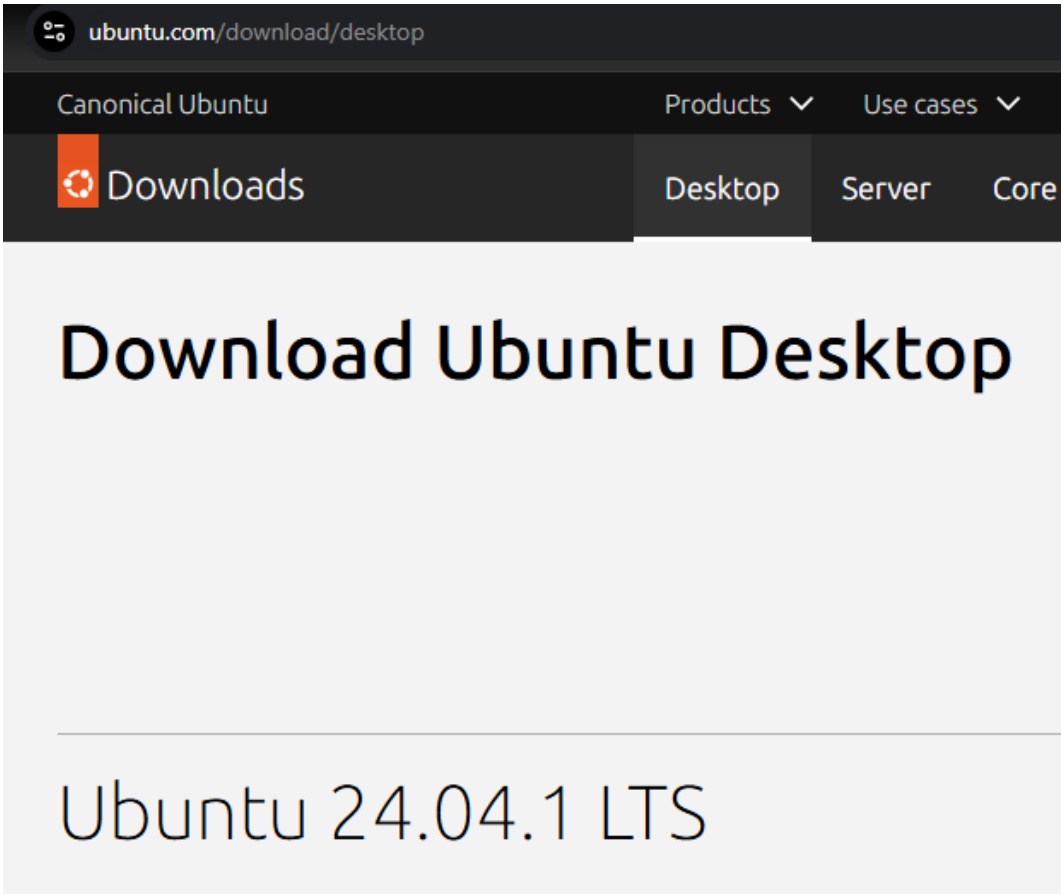
Create and run c codes in Windows or Mac machine.
MacOs already has Unix inside, but gcc may be installed.
With Windows you can use e.g some virtual machine (WmWare workstation is one choice).
You can install VMware and maybe Ubuntu package.

Example here



Create a short demoapplication using Linux or Unix. Show steps and results.

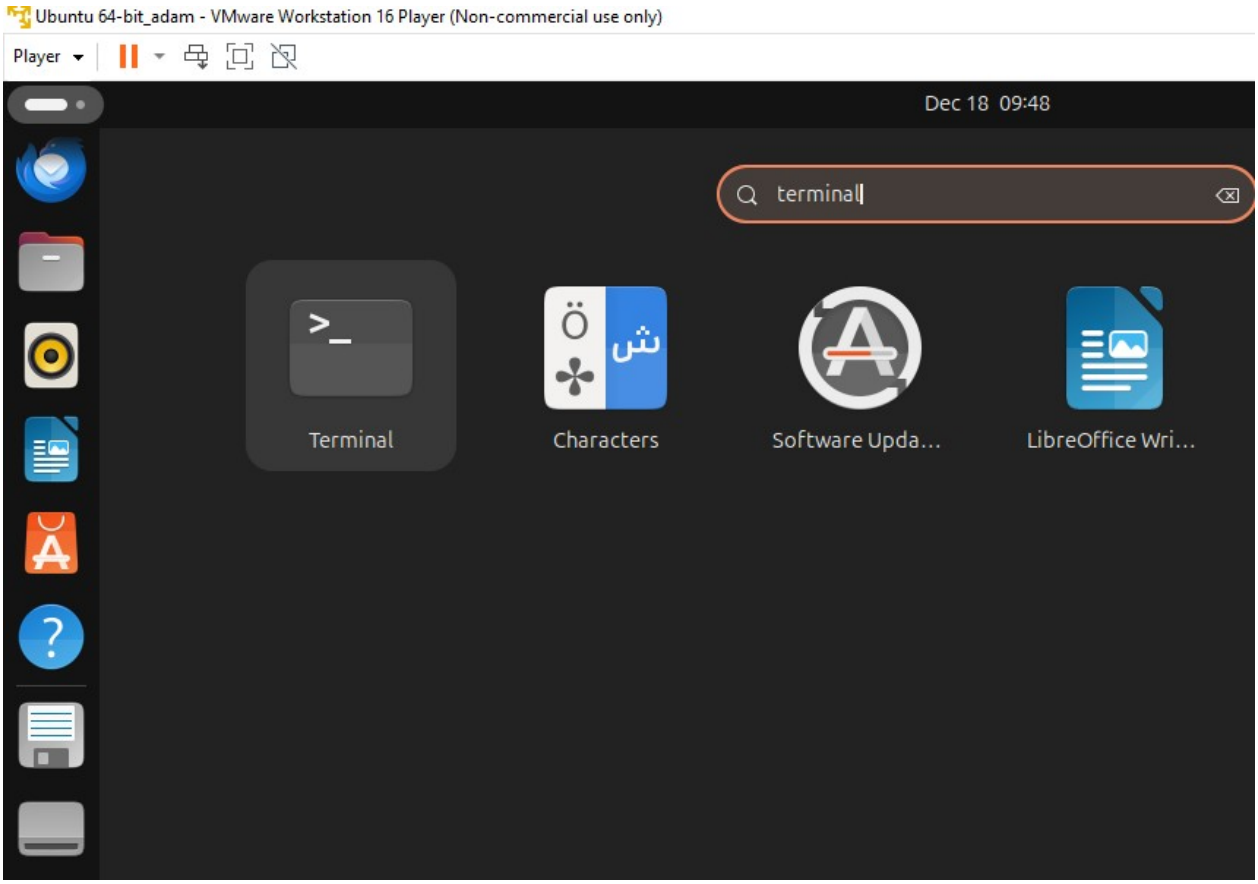
Solution
Follow these steps:
Download Linux package (.ISO package can be download vie net).



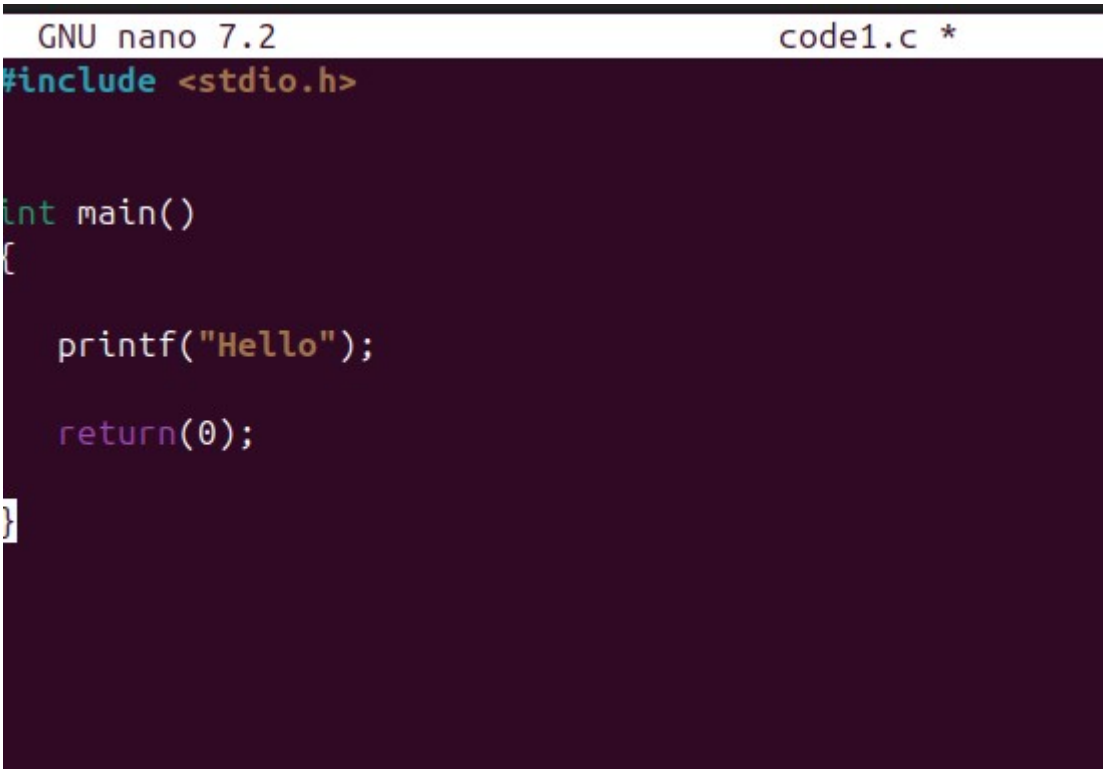
Create a new Virtual Machine

Start it

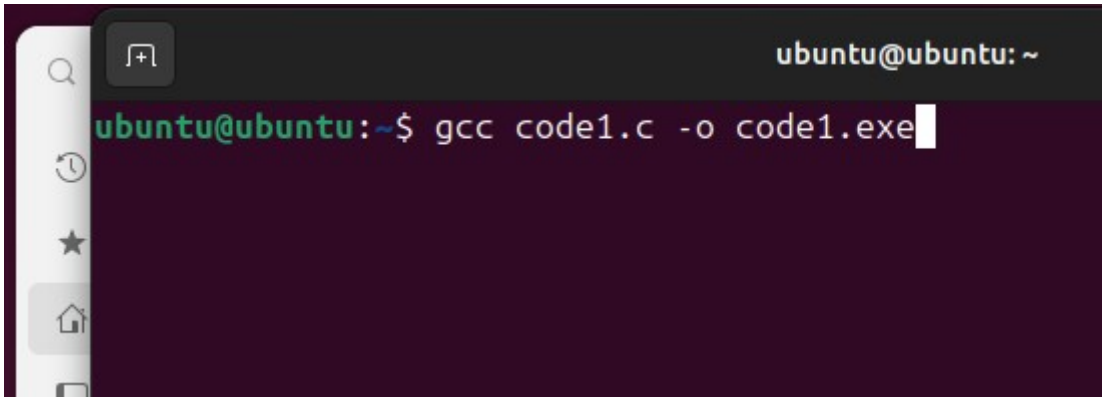
Open terminal.



If needed create a folder for c codes.
Use e.g. nano editor to create a code.

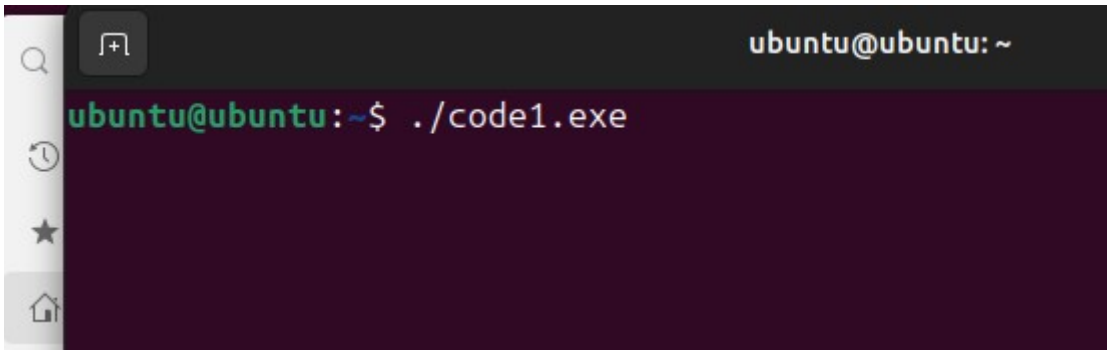


Compile the codefile (gcc).



Add rights to execute the file using chmod command if needed.

Run the file.



Try to do it yourself!!

Task 100

Generate Fibonacci value using a recursive function and non recursive function.

Solution

```
#include <stdio.h>

// 0, 1, 1, 2, 3, 5, 8, 13, 21, 34

int fibo1(int n)
{
    if (n == 0)
      return 0;
    else if (n == 1)
      return 1;
    else
```

```
      return fibo1(n-1) + fibo1(n-2);
}


int fibo2(int n)
{
    int fibs[10];
    fibs[0] = 0;
    fibs[1] = 1;
    int i;

    for (i = 2; i <= n; i++)
      fibs[i] = fibs[i-1] + fibs[i-2];

   return fibs[n];
}

int main()
{
  printf("%d \n", fibo1(5));

  printf("%d \n", fibo2(5));
  return 0;
}
```

SO this is it!

This ebook uses c language.

But later we use also C#, JavaScript and Python, too!

I hope you can give me comments: how to improve this book?

Thank You!