Name: Neptun:

If a code part does not lead closer to the solutions, that is 0, even if parts are close or same as in the solution, or any good solution

use absolutely wrong algorithm results in 0 points, as the incorrect use of elements (e.g. repetition instead of loop, lot of variables instead of array)

Syntax errors generate -1 or max -2 points of each type (even if it repeated many times in a task), if the meaning is not changed. E.G.: (no ; after declarations and operations does not change the meaning, missing {} after loop head changes the meaning.)

Name: Neptun:

Task 1: Write a program, which counts the number of alphabetic characters!

Write a program that reads one line from the keyboard and calculates the number of alphabetic letters.

Example:
IN: Alabama 12.
OUT:7.

```
#include <stdio.h>

int/void main(void){


int n=0; // 1p (initialization somewhere, and char declaration) if the algorithm after is good

char c;

scanf("%c",&c); //2p even if c initialized with a not alpha e.g. c='1';

while(c!='\n'){  //1p for loop and condition

        if(c>'a'&&c<'z' || c>'A'&&c<'Z') //2p each if the connection is right

                n++;

        scanf("%c",&c); //1p if at the right place

}

printf("%d",n); //1p

return;



}
```

Name: Neptun:

Task 2: Write a program, which gives back the statistics of the result of this midterm:

The mark boundaries of this midterm are 16-22-28-34 for marks 2, 3, 4 and 5, respectively. Your task is to write a program, which continuously reads the points terminated by any negative number (e.g.:-1). After the termination, the program prints out the number of students who received 1,2,3,4 and 5 marks! Note: The number of students is arbitrary.

Example:
IN: 16 25 40 40
OUT: 1:0 2:1 3:1 4:0 5:2

```
#include <stdio.h>

int/void main(void){


int a[]={0,0,0,0,0} //1p for all declarations; 1p for initialization

int n;

scanf(“%d”,&n); //1p

while(n>-1){ //3p: 1p if the if-else structure is OK; 2p if the operations are OK.

        if(n<16) {a[0]++;}

        else if (n<22) {a[1]++;}

        else if(n<28) {a[2]++;}

        else if (n<34) {a[3]++;}

        scanf(“%d”,&n);//1p

}

for(int i=0,i<5; i++){ //1p for loop)

        printf(“%d:%d “,i,a[i]); //1p

return;

}
```

Name: Neptun:

Task 3: Determine the value of $\sqrt[3]{3}$ by using iteration!

The program should work with accuracy read from the user, but at least 0.001! The result is printed out with 0.001 accuracy. Do not use math.h!

```
#include <stdio.h>


int/void main(void){//they usually take take the stepping solution

double eps, value=0;//1p value not necessary 0.

scanf("%lf",&eps);//1p using %f→0p

if(eps>0.001) eps=0.001; //1p

while(value*value*value-3<eps) //3p: All needed multiplication, substr. and correct cond.

        value+=eps;//1p

printf("%f.3",value); //1p


}
```

Name: Neptun:

Task 4: Prime numbers with 1 as the first digit

Write a program that prints out all the prime numbers between 1 and 1000 with the first digit equal to 1!

Example:
IN:
OUT:11,13,17,19…,197,199…1009,1013

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

SOL1:

```
int/void main(void){
for(int i=10;i<20;i++) //1p
{          int isprime=1;//1p
           for(int j=2;j<i/2;j++)  //1p (can go to n or sqrt(n)
                      if(i%j==0) isprime=0; //1p
           if(isprime) printf("%d",i); //2p cond+printf
}
for(int i=100;i<200;i++) //2p for the correct bounds
{          int isprime=1;
           for(int j=2;j<i/2;j++)
                      if(i%j==0) isprime=0;
           if(isprime) printf("%d",i);
}
```

SOL2:

```
int/void main(void){
for(int i=10;i<1000;i++) //1p
{          int isprime=1; //1p
           for(int j=2;j<i/2;j++) //1p
                      if(i%j==0) isprime=0; ==1p
           if(i<100&&i/10==1||i>100&&i/100==1) //2p
                      if(isprime==1) printf("%d",i);//2p
}
```

Name: Neptun:

Task 5: Create a program that writes all values below the average of a 12 long double array to the screen! The element of the array is given by the user!

Example:

IN: 1.0 2.2 4.8 6 7 23 56 76 1 3 5.5 6.5
OUT: 1.0 2.2 4.8 6 7 1 3 5.5 6.5
The average of the 12 elements of ***a*** array is 16 in this case!

```
#include <stdio.h>

int/void main(void){

double a[12]; //1p

double avg,sum=0; //1p


for(int i=0; i<12; i++) //1p

        scanf("%lf,",&a[i]); //1p

for(int i=0;i<12;i++){ //1p

        sum+=a[i]; //1p

avg=sum/n;//1p

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

        if (a[i]<avg) //1p

                printf("%d ",a[i]); //1p

        }

return;



}
```