Team 1

FAF-181

Project “oracle”

Team composition:

In this team there are 6 members. They are the following: Catalin Turcan, Loredana Godorogea, Dina-Alexandrina Bizgu, Cristina Cocieri, Alex Latcovschi, Victoria Belinschi.

Team Roles.

**Turcan Catalin** – Researched the methods of interpolation including polynomial and linear. Created the mathematical equation and wrote the program from 0 to a working state.

**Alex Latcovschi** – Improved the success rate of the program by improving the algorithm with new means. He discovered that for certain sets of integers, the program outputs a negative result. He is also the one that fixed the bug. Helped with the creation of a simpler way to solve the problem.

**Loredana Godorogea** – Gave the program a more appealing design by adding spaces, comments and removing unnecessary piece of code.

**Cristian Cocieri** – Optimized the program from using over 1mb of RAM and 9ms of processing time, down to about 200kb and 6ms of processing time.

**Victoria Belinschi** – Tested our 16 different combinations of interpolation methods in order to see which one gave a better result.

**Dina-Alexandrina Bizgu –** Tested the final version of the program with just over 1000 sets of integers, both “true random” and pseudo-random ones, to ensure there are no errors in the program.

The Problem:

We are given a set of 100 integers that were generated accordingly to an algorithm. Our task was to predict, with a margin of error as small as possible, the next integer.

Restrictions:

We weren’t allowed to use any library other than the default one.

We were allowed to use only the interpolation and extrapolation algorithms.

We were recommended to use as little “if” commands as possible.

The method

Upon a lot of tests and research based on the interpolation we found the following methods to output good results.

1)We interpolate from each point to the next one and then we do the arithmetic mean of the 99 equations.

2) We interpolate from each point to the last one and then we do the arithmetic mean of the 99 equations.

3)We do the arithmetic mean of the first 2 methods (1) and (2).

4)We interpolate and then extrapolate accordingly and apply the mean of the both.

5)We interpolate according to the polynomial method.

Upon conducting over 1000 tests on each of these methods, we found out that even if, the polynomial method and applying harmonic, geometric, quadratic means had better results, for certain values there were problems, like dividing by zero, and applying square root of negative numbers. Therefore we chose the classic linear interpolation.

How our program works?

1)The program is given just over 200 variables. There are mostly “x” and “y” which mean the postion of the number on OX and OY accordingly.

2)Using “scanf” our program receives input from the user the 100 values.

3)There is only one huge equation, which is actually a sum of 99 equations and then dividing the sum by 99. Each equation is the interpolation from a point to the next one.

4)This equation actually predicts the f(x) and then substitutes x = 101 (the 101th integer). This allows the program to find the values of function in that position. We do this 99 times and then we do the average of all these equations.

5)Sometimes the program outputs negative numbers. In this case the program simply amplifies the number with (-1).

6)If the output is positive by default, using the “printf” function, the program outputs the result.

The code itself:

----------------------------------------------------------

The program is available online: <https://github.com/Akulav/interpolare/blob/master/codfinal.txt>

Older versions can be found on:

<https://github.com/Akulav/interpolare>

-----------------------------------------------------------------------------------------------

The complete code is available at the end of the report.

Examples:

Here are 4 sequences of integers that were created by “true random” means, offered by random.org Each sequences contains 101 numbers.

“1046 7516 9219 9834 2740 6051 6191 7405 8878 8457 7431 373 2205 7372 7621 1160 6255 7462 4865 3144 211 2827 7361 5614 9700 2378 1534 2054 7015 7171 2249 8211 7794 7105 2414 1005 5169 8399 4835 2464 2564 6228 6383 1542 7578 1208 5897 5894 9874 4177 4241 2802 7969 8538 5851 6680 7310 7697 2582 8359 5708 9560 7147 2604 2239 6258 8972 3256 2187 5668 7363 8756 3604 273 9766 5463 2420 8778 2624 3181 4523 1767 8875 4657 2865 5190 936 1725 2082 3024 2075 9933 1141 537 3857 9026 311 7777 3726 1839” . Correct answer – 9798. Predicted answer – 9059

“4830 5686 1448 4755 656 1082 3568 4988 6701 1220 4644 1176 2867 9848 2713 3599 6982 3821 5486 8806 7384 4722 723 3537 6412 9510 6606 4781 7272 6669 755 3551 7244 2052 8352 4140 9205 6583 8580 8085 1746 4529 1565 5333 7861 6455 5796 5378 2313 4005 6278 1485 4074 6351 6046 4124 3195 1039 2619 3078 3609 64 1442 7985 8110 7359 2706 7452 4020 4814 1298 5139 5365 8593 90 2745 9547 8864 4632 710 1562 8619 4873 9556 1385 9483 122 2 4105 5030 6985 1533 2066 828 496 8571 3626 3288 1317 3613” Correct answer – 4942. Predicted answer – 4239

“4398 2053 7424 3195 6349 9944 6737 3474 3093 8713 9444 3656 3810 9212 85 3125 8327 8339 1867 2410 4652 176 6751 5826 2837 7061 1242 620 7103 9872 1970 8105 4524 9693 4208 8599 350 1805 8404 1523 869 1657 3631 6071 890 7933 772 1541 6307 4086 9621 2845 5958 6210 3180 222 2080 9411 6679 586 9712 2251 1289 99 6627 3654 4438 5598 3639 192 6165 8844 5829 4639 8317 8418 216 7795 977 9251 7568 2485 5650 2017 7179 5570 3722 7929 1948 5442 127 1130 3726 6479 3147 3238 6809 8670 5727 5082”

Correct answer – 6321. Predicted answer – 5020

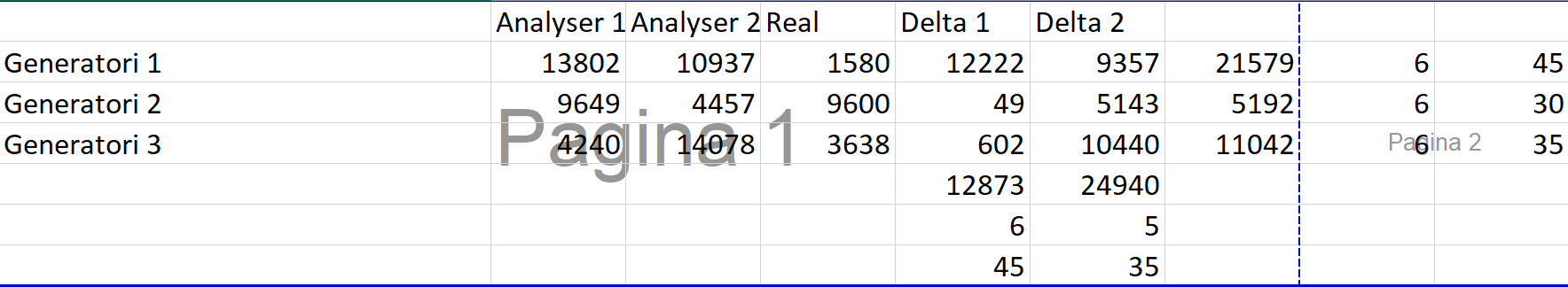
“619 8679 9149 4141 4774 3161 283 8476 3056 6270 2107 3396 1974 9798 3518 9462 6283 1338 5040 4773 3852 7388 4796 7987 2563 4913 3499 3515 222 394 7772 3064 8926 652 9502 3253 2004 1781 2219 8501 9649 1753 2704 94 2435 3979 3742 7002 3423 712 1701 3245 8089 6423 2549 5531 868 6387 9009 2492 7735 9560 6464 1231 1992 3048 9735 4509 4358 276 7228 2469 2951 5319 1843 6132 6188 3279 9109 2856 4147 3400 3557 9663 5704 3306 3556 8486 1321 2594 9985 1777 5366 929 7796 7722 4 95 166 8399”

Correct answer – 8942. Predicted answer – 8357

Analytics:

According to the 4 sequences on the previous page, our program outputs results within a max error of 21% . After doing other 1000 tests, we found out that the average error is 33%. We do believe, after many trial and error attempts, that our method has an error of only +-5% than the polynomial one, but it is 9% more reliable, since the polynomial method can result in numbers too high or too low. If the random values have duplicates, which happen in non pseudo-random algorithms, it implies division by zero, which automatically cant be calculated. This is why, according to statistics we created, our method of linear interpolation is the best.

By doing it simple from point to point, and applying a simple arithmetic mean, we got results that were at least 2X times more accurate than the other team in our group.



As you can see out team, Analyser 1, had a smaller error than the other team, according to “delta 1” and “delta 2”.

Conclusion:

Our team created multiple mathematical algorithms and then we tested each to see which is the best. Some were better, but non-reliable and vice-versa. We chose the middle option that should work 100% of the time, with a relative small error. This is the holy grail of predicting the next number in a sequence.

Even if not each member worked that hard in making the program itself, we do believe that each role is important. It doesn’t matter how many developers are there, if there is no one to organize the code, optimize it, or test it, the team won’t work.

The program:

#include <stdio.h>

int **main**()

{

//Se declara variabilele. y - pozitia punctului pe axa oy, x - pozitia punctului pe ox, h - x cu coordonata (101;y), ytotal - medie aritmetica a 99 de

//ecuatii pentru interpolarea liniara

int ytotal,ytotal1,yclasic,yclasic1,yclasic2;

int y1 ,y2 ,y3 ,y4 ,y5 ,y6 ,y7 ,y8 ,y9 ,y10;

int y11,y12,y13,y14,y15,y16,y17,y18,y19,y20;

int y21,y22,y23,y24,y25,y26,y27,y28,y29,y30;

int y31,y32,y33,y34,y35,y36,y37,y38,y39,y40;

int y41,y42,y43,y44,y45,y46,y47,y48,y49,y50;

int y51,y52,y53,y54,y55,y56,y57,y58,y59,y60;

int y61,y62,y63,y64,y65,y66,y67,y68,y69,y70;

int y71,y72,y73,y74,y75,y76,y77,y78,y79,y80;

int y81,y82,y83,y84,y85,y86,y87,y88,y89,y90;

int y91,y92,y93,y94,y95,y96,y97,y98,y99,y100;

int x1 , x2, x3, x4, x5, x6, x7, x8, x9,x10;

int x11,x12,x13,x14,x15,x16,x17,x18,x19,x20;

int x21,x22,x23,x24,x25,x26,x27,x28,x29,x30;

int x31,x32,x33,x34,x35,x36,x37,x38,x39,x40;

int x41,x42,x43,x44,x45,x46,x47,x48,x49,x50;

int x51,x52,x53,x54,x55,x56,x57,x58,x59,x60;

int x61,x62,x63,x64,x65,x66,x67,x68,x69,x70;

int x71,x72,x73,x74,x75,x76,x77,x78,x79,x80;

int x81,x82,x83,x84,x85,x86,x87,x88,x89,x90;

int x91,x92,x93,x94,x95,x96,x97,x98,x99,x100;

int h=101;

//Se cere introducerea celor 100 de numere. Si se declara implicit valoarea celor 100 de x, cu pozitiile lor corespunzatoare pe ox.

printf("Introdu prin spatii cele 100 de numere\n");

scanf("%d%d%d%d%d%d%d%d%d%d%d%d%d%d%d%d%d%d%d%d"

"%d%d%d%d%d%d%d%d%d%d%d%d%d%d%d%d%d%d%d%d"

"%d%d%d%d%d%d%d%d%d%d%d%d%d%d%d%d%d%d%d%d"

"%d%d%d%d%d%d%d%d%d%d%d%d%d%d%d%d%d%d%d%d"

"%d%d%d%d%d%d%d%d%d%d%d%d%d%d%d%d%d%d%d%d"

,&y1,&y2,&y3,&y4,&y5,&y6,&y7,&y8,&y9,&y10

,&y11,&y12,&y13,&y14,&y15,&y16,&y17,&y18,&y19,&y20

,&y21,&y22,&y23,&y24,&y25,&y26,&y27,&y28,&y29,&y30

,&y31,&y32,&y33,&y34,&y35,&y36,&y37,&y38,&y39,&y40

,&y41,&y42,&y43,&y44,&y45,&y46,&y47,&y48,&y49,&y50

,&y51,&y52,&y53,&y54,&y55,&y56,&y57,&y58,&y59,&y60

,&y61,&y62,&y63,&y64,&y65,&y66,&y67,&y68,&y69,&y70

,&y71,&y72,&y73,&y74,&y75,&y76,&y77,&y78,&y79,&y80

,&y81,&y82,&y83,&y84,&y85,&y86,&y87,&y88,&y89,&y90

,&y91,&y92,&y93,&y94,&y95,&y96,&y97,&y98,&y99,&y100);

x1=1;x2=2;x3=3;x4=4;x5=5;x6=6;x7=7;x8=8;x9=9;x10=10;x11=11;x12=12;x13=13;x14=14;x15=15;

x16=16;x17=17;x18=18;x19=19;x20=20;x21=21;x22=22;x23=23;x24=24;x25=25;

x26=26;x27=27;x28=28;x29=29;x30=30;x31=31;x32=32;x33=33;x34=34;x35=35;x36=36;x37=37;x38=38;x39=39;x40=40;x41=41;x42=42;x43=43;x44=44;x45=45;x46=46;x47=47;

x48=48;x49=49;x50=50;x51=51;x52=52;x53=53;x54=54;x55=55;x56=56;x57=57;x58=58;x59=59;x60=60;x61=61;x62=62;x63=63;x64=64;x65=65;x66=66;x67=67;x68=68;x69=69;x70=70;

x71=71;x72=72;x73=73;x74=74;x75=75;x76=76;x77=77;x78=78;x79=79;x80=80;x81=81;x82=82;x83=83;x84=84;x85=85;x86=86;x87=87;x88=88;x89=89;x90=90;x91=91;x92=92;

x93=93;x94=94;x95=95;x96=96;x97=97;x98=98;x99=99;x100=100;

//Se calculeaza media interpolarii liniare de la punct la punct.

ytotal=(((y1\*(x2-h)+y2\*(h-x1))/(x2-x1))+((y2\*(x3-h)+y3\*(h-x2))/(x3-x2))+

((y3\*(x4-h)+y4\*(h-x3))/(x4-x3))+((y4\*(x5-h)+y5\*(h-x4))/(x5-x4))+

((y5\*(x6-h)+y6\*(h-x5))/(x6-x5))+((y6\*(x7-h)+y7\*(h-x6))/(x7-x6))+

((y7\*(x8-h)+y8\*(h-x7))/(x8-x7))+((y8\*(x9-h)+y9\*(h-x8))/(x9-x8))+

((y9\*(x10-h)+y10\*(h-x9))/(x10-x9))+((y10\*(x11-h)+y11\*(h-x10))/(x11-x10))+

((y11\*(x12-h)+y12\*(h-x11))/(x12-x11))+((y12\*(x13-h)+y13\*(h-x12))/(x13-x12))+

((y13\*(x14-h)+y14\*(h-x13))/(x14-x13))+((y14\*(x15-h)+y15\*(h-x14))/(x15-x14))+

((y15\*(x16-h)+y16\*(h-x15))/(x16-x15))+((y16\*(x17-h)+y17\*(h-x16))/(x17-x16))+

((y17\*(x18-h)+y18\*(h-x17))/(x18-x17))+((y18\*(x19-h)+y19\*(h-x18))/(x19-x18))+

((y19\*(x20-h)+y20\*(h-x19))/(x20-x19))+((y20\*(x21-h)+y21\*(h-x20))/(x21-x20))+

((y21\*(x22-h)+y22\*(h-x21))/(x22-x21))+((y22\*(x23-h)+y23\*(h-x22))/(x23-x22))+

((y23\*(x24-h)+y24\*(h-x23))/(x24-x23))+((y24\*(x25-h)+y25\*(h-x24))/(x25-x24))+

((y25\*(x26-h)+y26\*(h-x25))/(x26-x25))+((y26\*(x27-h)+y27\*(h-x26))/(x27-x26))+

((y27\*(x28-h)+y28\*(h-x27))/(x28-x27))+((y28\*(x29-h)+y29\*(h-x28))/(x29-x28))+

((y29\*(x30-h)+y30\*(h-x29))/(x30-x29))+((y30\*(x31-h)+y31\*(h-x30))/(x31-x30))+

((y31\*(x32-h)+y32\*(h-x31))/(x32-x31))+((y32\*(x33-h)+y33\*(h-x32))/(x33-x32))+

((y33\*(x34-h)+y34\*(h-x33))/(x34-x33))+((y34\*(x35-h)+y35\*(h-x34))/(x35-x34))+

((y35\*(x36-h)+y36\*(h-x35))/(x36-x35))+((y36\*(x37-h)+y37\*(h-x36))/(x37-x36))+

((y37\*(x38-h)+y38\*(h-x37))/(x38-x37))+((y38\*(x39-h)+y39\*(h-x38))/(x39-x38))+

((y39\*(x40-h)+y40\*(h-x39))/(x40-x39))+((y40\*(x41-h)+y41\*(h-x40))/(x41-x40))+

((y41\*(x42-h)+y42\*(h-x41))/(x42-x41))+((y42\*(x43-h)+y43\*(h-x42))/(x43-x42))+

((y43\*(x44-h)+y44\*(h-x43))/(x44-x43))+((y44\*(x45-h)+y45\*(h-x44))/(x45-x44))+

((y45\*(x46-h)+y46\*(h-x45))/(x46-x45))+((y46\*(x47-h)+y47\*(h-x46))/(x47-x46))+

((y47\*(x48-h)+y48\*(h-x47))/(x48-x47))+((y48\*(x49-h)+y49\*(h-x48))/(x49-x48))+

((y49\*(x50-h)+y50\*(h-x49))/(x50-x49))+((y50\*(x51-h)+y51\*(h-x50))/(x51-x50))+

((y51\*(x52-h)+y52\*(h-x51))/(x52-x51))+((y52\*(x53-h)+y53\*(h-x52))/(x53-x52))+

((y53\*(x54-h)+y54\*(h-x53))/(x54-x53))+((y54\*(x55-h)+y55\*(h-x54))/(x55-x54))+

((y55\*(x56-h)+y56\*(h-x55))/(x56-x55))+((y56\*(x57-h)+y57\*(h-x56))/(x57-x56))+

((y57\*(x58-h)+y58\*(h-x57))/(x58-x57))+((y58\*(x59-h)+y59\*(h-x58))/(x59-x58))+

((y59\*(x60-h)+y60\*(h-x59))/(x60-x59))+((y60\*(x61-h)+y61\*(h-x60))/(x61-x60))+

((y61\*(x62-h)+y62\*(h-x61))/(x62-x61))+((y62\*(x63-h)+y63\*(h-x62))/(x63-x62))+

((y63\*(x64-h)+y64\*(h-x63))/(x64-x63))+((y64\*(x65-h)+y65\*(h-x64))/(x65-x64))+

((y65\*(x66-h)+y66\*(h-x65))/(x66-x65))+((y66\*(x67-h)+y67\*(h-x66))/(x67-x66))+

((y67\*(x68-h)+y68\*(h-x67))/(x68-x67))+((y68\*(x69-h)+y69\*(h-x68))/(x69-x68))+

((y69\*(x70-h)+y70\*(h-x69))/(x70-x69))+((y70\*(x71-h)+y71\*(h-x70))/(x71-x70))+

((y71\*(x72-h)+y72\*(h-x71))/(x72-x71))+((y72\*(x73-h)+y73\*(h-x72))/(x73-x72))+

((y73\*(x74-h)+y74\*(h-x73))/(x74-x73))+((y74\*(x75-h)+y75\*(h-x74))/(x75-x74))+

((y75\*(x76-h)+y76\*(h-x75))/(x76-x75))+((y76\*(x77-h)+y77\*(h-x76))/(x77-x76))+

((y77\*(x78-h)+y78\*(h-x77))/(x78-x77))+((y78\*(x79-h)+y79\*(h-x78))/(x79-x78))+

((y79\*(x80-h)+y80\*(h-x79))/(x80-x79))+((y80\*(x81-h)+y81\*(h-x80))/(x81-x80))+

((y81\*(x82-h)+y82\*(h-x81))/(x82-x81))+((y82\*(x83-h)+y83\*(h-x82))/(x83-x82))+

((y83\*(x84-h)+y84\*(h-x83))/(x84-x83))+((y84\*(x85-h)+y85\*(h-x84))/(x85-x84))+

((y85\*(x86-h)+y86\*(h-x85))/(x86-x85))+((y86\*(x87-h)+y87\*(h-x86))/(x87-x86))+

((y87\*(x88-h)+y88\*(h-x87))/(x88-x87))+((y88\*(x89-h)+y89\*(h-x88))/(x89-x88))+

((y89\*(x90-h)+y90\*(h-x89))/(x90-x89))+((y90\*(x91-h)+y91\*(h-x90))/(x91-x90))+

((y91\*(x92-h)+y92\*(h-x91))/(x92-x91))+((y92\*(x93-h)+y93\*(h-x92))/(x93-x92))+

((y93\*(x94-h)+y94\*(h-x93))/(x94-x93))+((y94\*(x95-h)+y95\*(h-x94))/(x95-x94))+

((y95\*(x96-h)+y96\*(h-x95))/(x96-x95))+((y96\*(x97-h)+y97\*(h-x96))/(x97-x96))+

((y97\*(x98-h)+y98\*(h-x97))/(x98-x97))+((y98\*(x99-h)+y99\*(h-x98))/(x99-x98))+

((y99\*(x100-h)+y100\*(h-x99))/(x100-x99)))/100;

//Se afiseaza raspunsul

if (ytotal<0) {printf("valoarea extrapolata",ytotal\*(-1));};

printf("\n\n Valoarea aproximativa interpolata a numarului in pozitia 101 este (yclasic): %d", ytotal);

return 0;

}