1.  Elemente fundamentale ale limbajului C++ (operatori, expresii, variabile)

Sa se determine valoarea functieti urmatoare pentru x, numar real citit de la tastatura.

F:R🡪R, f(x)

#include <iostream>

using namespace std;

double x, b, a, c;

int main()

{

cout << "Se da numarul x: ";

cin >> x;

a = 5 \* (x \* x);

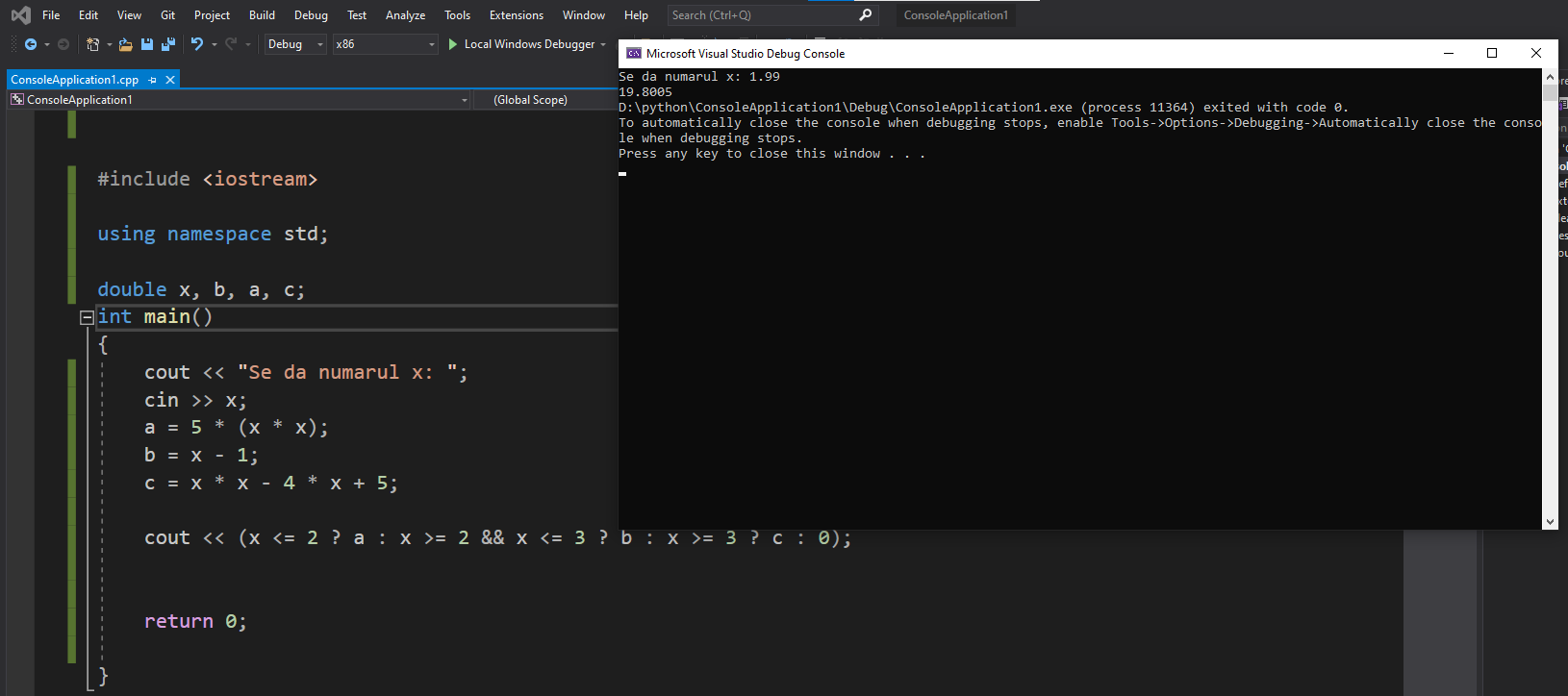
b = x - 1;

c = x \* x - 4 \* x + 5;

cout << (x <= 2 ? a : x >= 2 && x <= 3 ? b : x >= 3 ? c : 0);

return 0;

}



2. Instructiuni ale limbajului

Se citeste un numar natural n. Afisati factorul prim care apare la puterea cea mai mare in descompunerea in factori primi a lui n.  
Daca mai multi factori apar ca acea putere maxima, atunci se va afisa cel mai mare dintre ei.  
Exemplu: n=36 => 3 (din 2 la 2 si 3 la 2), iar n=40 =>2 (din 2 la 3 si 5 la 1)

#include <iostream>

using namespace std;

int main()

{

int n, d = 2, emax = 1, dmax;

cin >> n;

while (n > 1)

if (n % d == 0)

{

int e = 0;

while (n % d == 0) { e++; n = n / d; }

if (e >= emax) { emax = e; dmax = d; }

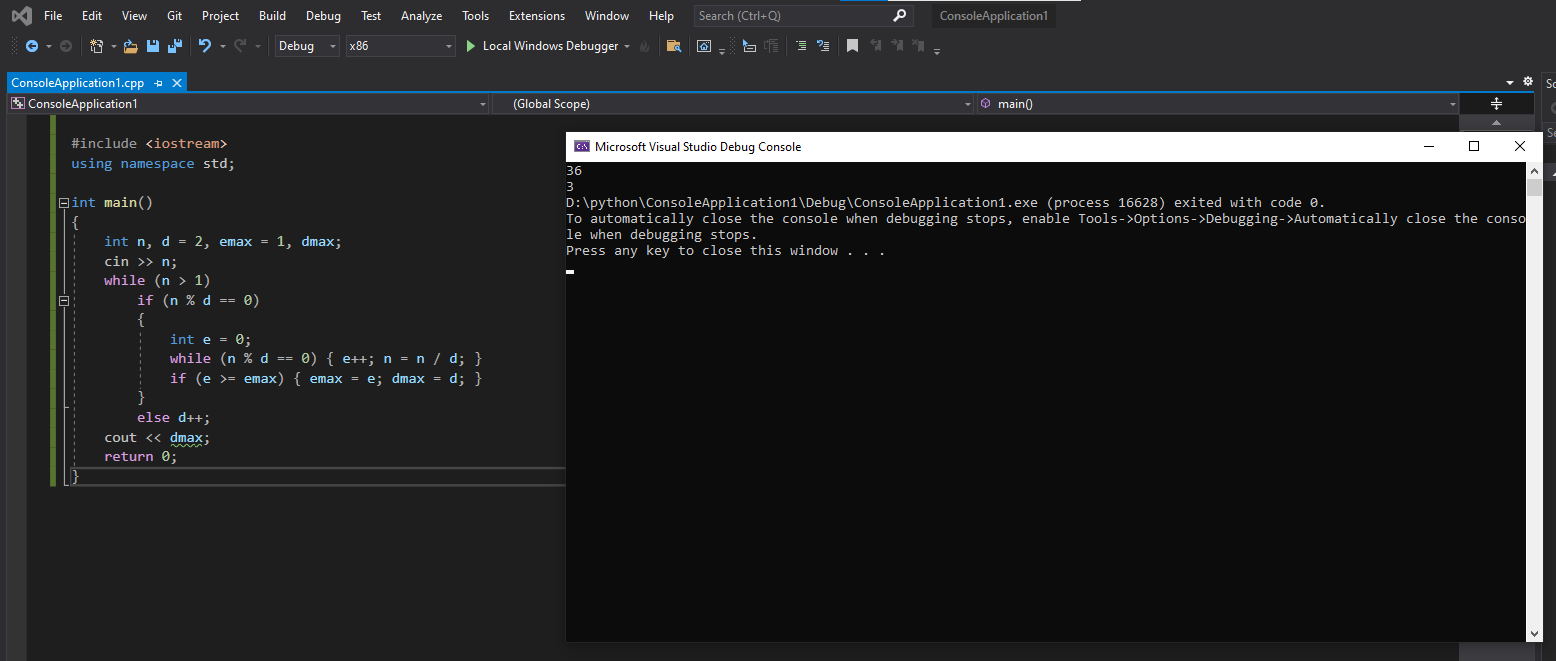
}

else d++;

cout << dmax;

return 0;

}



3. Pointeri si referinte

Se dă un număr natural n. Să se descompună în sumă cu număr minim de termeni ai şirului lui Fibonacci.

#include <iostream>

using namespace std;

int fib(int n)

{

int f1 = 1, f2 = 1, f3;

while (f1 + f2 <= n)

{

f3 = f1 + f2;

f1 = f2;

f2 = f3;

}

return f2;

}

int main()

{

int n;

cin >> n;

while (n != 0)

{

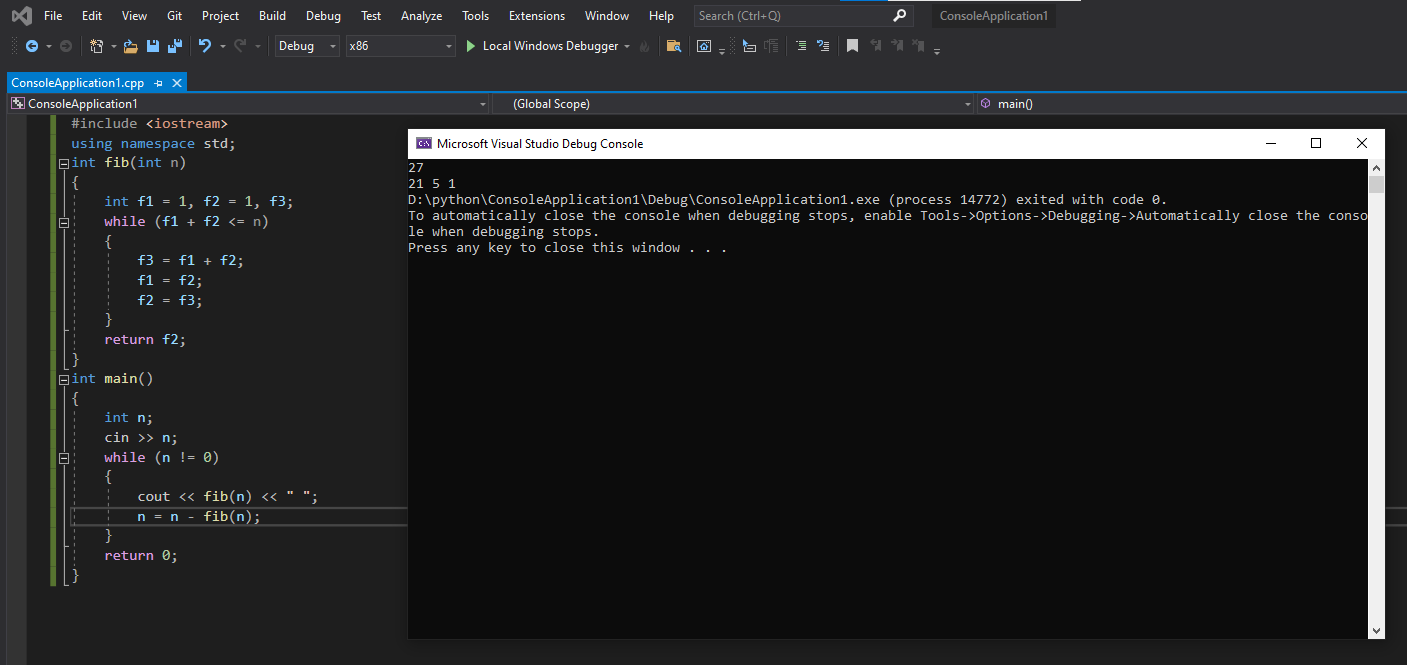
cout << fib(n) << " ";

n = n - fib(n);

}

return 0;

}



4. Functii definite de utilizator

Se citeste un vector cu n elemente numere naturale. Sa se afiseze toate perechile de elemente din vector care sunt formate din elemente prime intre ele.  
Ex: Pentru vectorul 34 22 15 77 50 se vor afisa perechile:  
34 15  
34 77  
22 15  
15 77  
77 50

#include<fstream.h>

ifstream fin("p.in");

ofstream fout("p.out");

void citire(int a[100], int& n)

{

int i;

fin >> n;

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

fin >> a[i];

}

int cmmdc(int x, int y)

{

while (x != y)

if (x > y)

x = x - y;

else y = y - x;

return x;

}

int main()

{

int a[100], n, i, j;

citire(a, n);

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

for (j = i + 1; j <= n; j++)

if (cmmdc(a[i], a[j]) == 1)

fout << a[i] << " " << a[j] << endl;

}

5. Array-uri

Se citeste un vector cu n elemente numere intregi, n fiind cel mult egal cu 1000. Parcurgeti si afisati elementele vectorului astfel: primul, ultimul, al doilea, penultimul, ...  
Exemplu: Pentru n=7 si vectorul cu elementele {5, 4, 5, 2, 7, 6, 5} se vor afisa numerele: 5 5 4 6 5 7 2

#include <iostream>

using namespace std;

int main()

{

int A[1001], n;

cin >> n;

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

cin >> A[i];

int i = 1, j = n;

while (i < j)

{

cout << A[i] << " " << A[j] << " ";

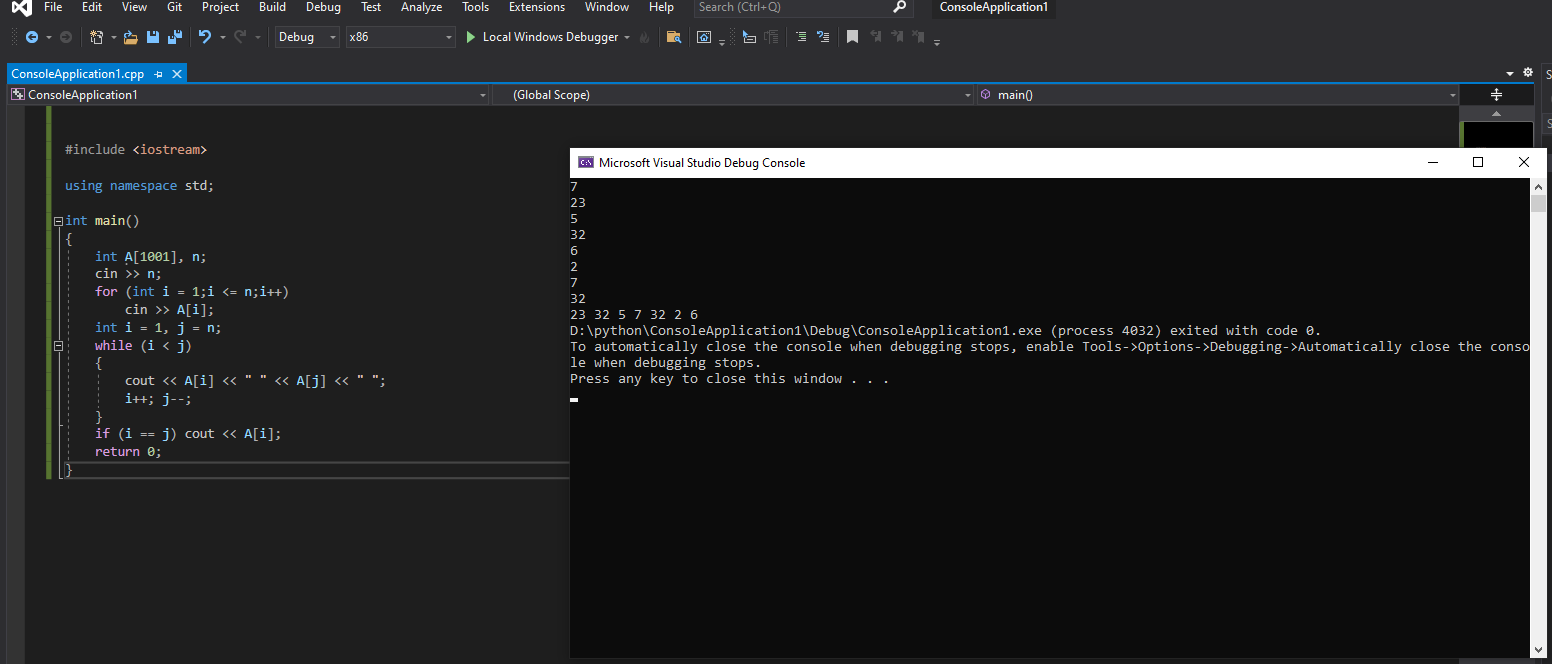
i++; j--;

}

if (i == j) cout << A[i];

return 0;

}



6. Matrice multi-dimensionale

Se citeste un numar natural n (n<=100). Construiti si afisati o matrice patratica cu n linii si n coloane dupa modelele de mai jos.  
Pentru n=7 se obtine matricea:  
1 2 3 4 3 2 1  
2 3 4 5 4 3 2  
3 4 5 6 5 4 3  
4 5 6 7 6 5 4  
3 4 5 6 5 4 3  
2 3 4 5 4 3 2  
1 2 3 4 3 2 1  
Pentru n=8 se obtine matricea:  
1 2 3 4 4 3 2 1  
2 3 4 5 5 4 3 2  
3 4 5 6 6 5 4 3  
4 5 6 7 7 6 5 4  
4 5 6 7 7 6 5 4  
3 4 5 6 6 5 4 3  
2 3 4 5 5 4 3 2  
1 2 3 4 4 3 2 1

#include <iostream>

using namespace std;

int main()

{

int n, A[101][101];

cin >> n;

for (int i = 1;i <= (n + 1) / 2;i++)

for (int j = 1;j <= (n + 1) / 2;j++)

A[i][j] = A[i][n + 1 - j] = A[n + 1 - i][j] = A[n + 1 - j][n + 1 - i] = i + j - 1;

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

{

for (int j = 1;j <= n;j++)

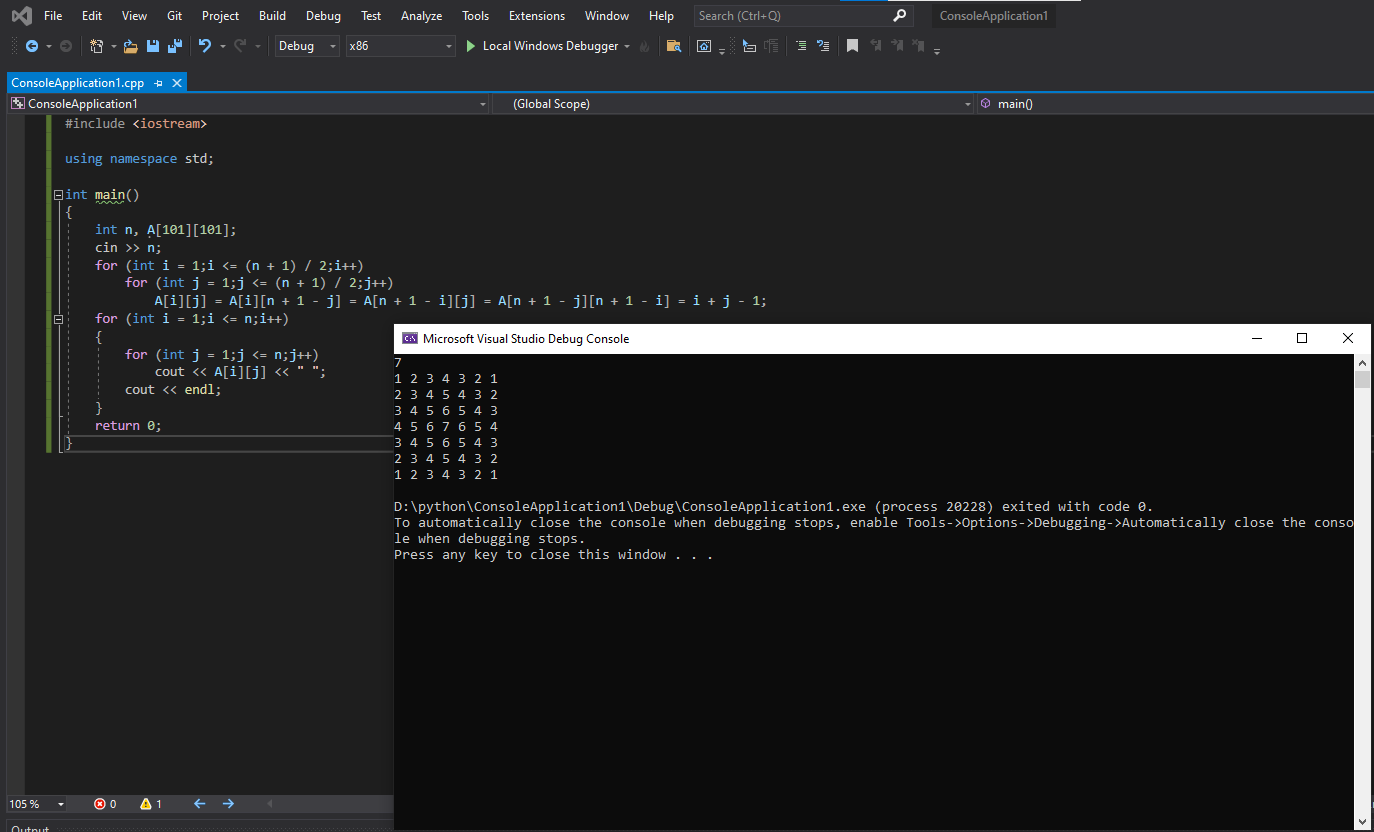
cout << A[i][j] << " ";

cout << endl;

}

return 0;

}



7. Inregistrari (structuri si uniuni)

Definiti un tip de date pentru memorarea lungimii si latimii unui dreptunghi. Cititi un numar natural n si apoi n perechi de numere intregi reprezentand n dreptunghiuri.  
a) Afisati laturile dreptunghiului cu arie maxima.  
b) Afisati laturile dreptunghiului cu perimetrul minim.

#include<fstream>

using namespace std;

ifstream fin("date.in");

ofstream fout("date.out");

struct dreptunghi

{

int L, l, a, p;

};

void citire(dreptunghi& d)

{

fin >> d.L >> d.l;

d.a = d.L \* d.l;

d.p = 2 \* (d.l + d.L);

}

void afisare(dreptunghi d)

{

fout << d.L << " " << d.l << endl;

}

int main()

{

int n, i;

dreptunghi d, max, min;

fin >> n;

citire(d);

max = min = d;

for (i = 2;i <= n;i++)

{

citire(d);

if (d.a > max.a) max = d;

if (d.p < min.p) min = d;

}

afisare(max);

afisare(min);

fin.close();

fout.close();

return 0;

}

8. Fisiere

Fisierul text bac.txt contine, pe o singura linie, cel mult 1000 de numere naturale nenule, numerele fiind separate prin cate un spatiu. Scrieti un program C/C++ care citeste de la tastatura un numar natural nenul n si numerele din fisierul bac.txt si care afiseaza pe ecran, separate prin cate un spatiu, toate numerele din fisier care sunt divizibile cu n. Daca fisierul nu contine niciun astfel de numar, atunci se va afisa pe ecran mesajul NU EXISTA.  
Exemplu: daca fisierul bac.txt contine numerele: 3 100 40 70 25 5 80 6 3798,  
pentru n=10 atunci pe ecran se va afisa: 100 40 70 80

#include<fstream>

using namespace std;

ifstream f("bac.txt");

int main()

{

int n, x, k = 0;

cin >> n;

while (f >> x)

if (x % n == 0) {

cout << x << " ";

k++;

}

if (k == 0) cout << "NU EXISTA";

return 0;

}

9. Siruri de caractere

Se consideră două şiruri de caractere a, de lungime na şi b, de lungime nb. Şirul a este numit prefix al şirului b dacă na≤nb şi subşirul lui b determinat de primele sale na caractere coincide cu a. Şirul a este numit sufix al şirului b dacă na≤nb şi subşirul lui b determinat de ultimele sale na caractere coincide cu a. Scrieţi un program C/C++ care citeşte de la tastatură două cuvinte, fiecare fiind format din cel mult 30 de caractere, numai litere mici ale alfabetului englez. După primul cuvânt se tastează Enter. Programul afişează pe ecran, separate prin câte un spaţiu, toate sufixele primului cuvânt care sunt prefixe ale celui de al doilea cuvânt, ca în exemplu. Dacă nu există astfel de subşiruri, se afişează pe ecran mesajul NU EXISTA. Exemple: dacă se citesc, în ordine,

cuvintele rebele elegant

se afişează (nu neapărat în această ordine) subşirurile:

e ele

iar dacă se citesc, în ordine, cuvintele

strai rai

se afişează subşirul: rai

#include <iostream>

#include <cstring>

using namespace std;

int main()

{

char a[31], b[31];

int gasit = 0;

cin >> a >> b;

for (int i = strlen(a) - 1;i > 0;i--)

if (strstr(b, a + i) == b) //daca sufixul a+i e la inceputul lui b

{

cout << a + i << " ";//afisez sufixul

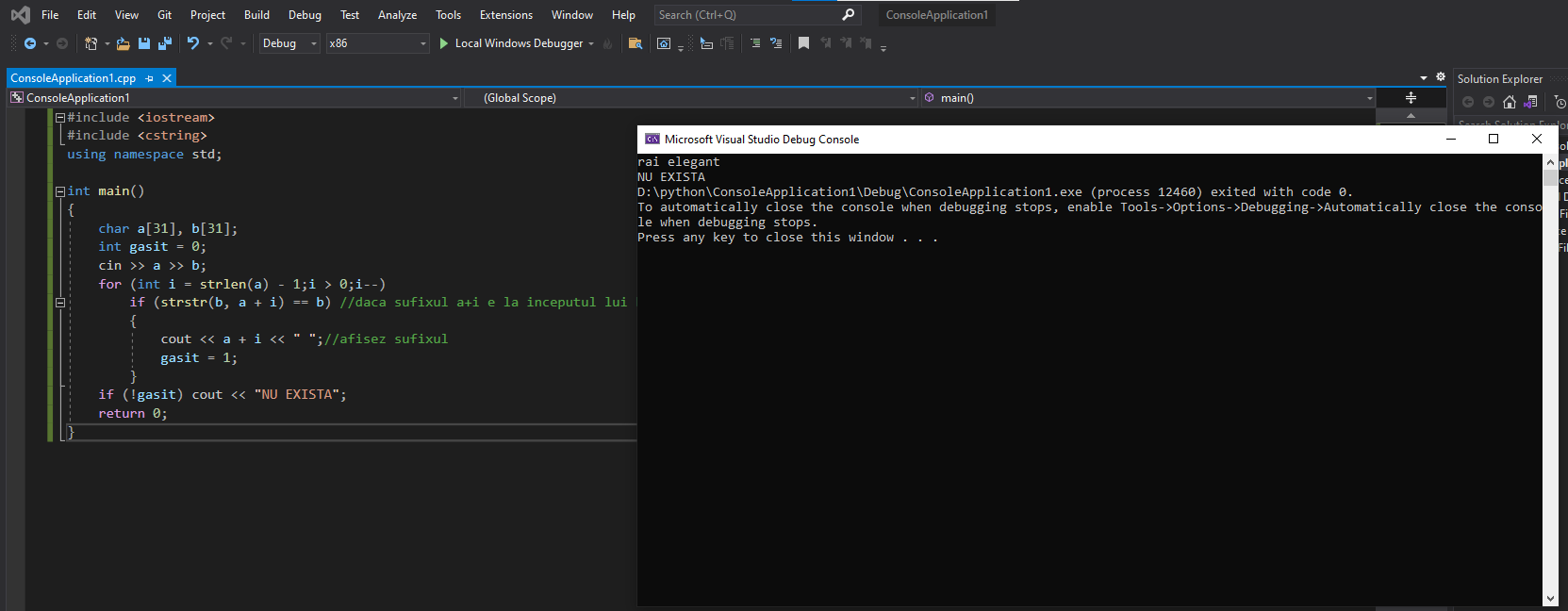
gasit = 1;

}

if (!gasit) cout << "NU EXISTA";

return 0;

}



10. Recursivitate

Se citeste un numar natural n. Sa se descompuna numarul n in toate modurile ca suma de doua numere a si b care au proprietatea ca b este rasturnatul lui a. Se vor scrie si folosi doua functii recursive pentru:  
- calculul rasturnatului  
- descompunerea ceruta  
Exemplu: numarul 787 se descompune ca 146+641, 245+542 sau 344+443

#include <iostream>

using namespace std;

int rast(int n, int r)

{

if (n == 0) return r;

else return rast(n / 10, r \* 10 + n % 10);

}

void desc(int x, int i)

{

if (i <= x / 2)

{

if (i + rast(i, 0) == x) cout << i << "+" << rast(i, 0) << endl;

desc(x, i + 1);

}

}

int main()

{

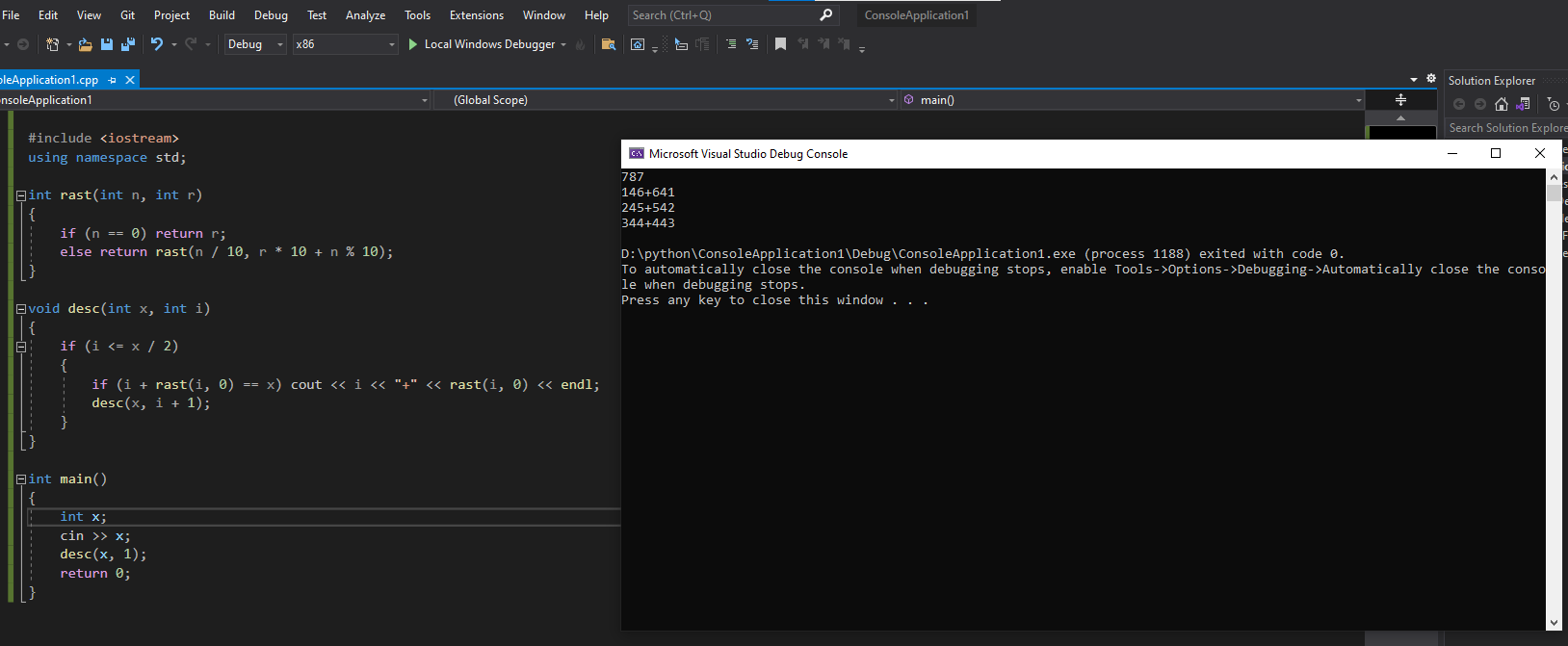
int x;

cin >> x;

desc(x, 1);

return 0;

}



11. Structuri dinamice de date (liste inlantuite, arbori)

Se considera o lista liniara simplu inlantuita cu nodurile definite astfel:  
struct Node  
{  
int key; //informatia memorata (utila)  
Node\* next; //legatura spre nodul urmator  
};  
Scrieti o functie care sa ordoneze crescator informatiile din lista. Functia va primi ca paramentru adresa primului nod al listei.

void SortAscBubble(Node\* first)

{//ordonez crescator cu metoda bulelor

int gata;

do

{

gata = 1;

for (Node\* p = first; p->next; p = p->next)//parcurg pana in fata ultimului

if (p->key > p->next->key)//p si p->next nu au informatiile in ordine

{// interchimb informatiile din cele 2 noduri

int aux = p->key;

p->key = p->next->key;

p->next->key = aux;

gata = 0;

}

} while (!gata);

}

Se da un numar natural n. Construiti un arbore binar complet cu varfurile 1,2...n astfel incat in urma parcurgerii pe nivele sa fie afisate valorile 1,2...n.  
Se vor afisa vectorii S si D.

#include<fstream>

#include<cmath>

using namespace std;

ifstream fin("date.in");

ofstream fout("date.out");

int i, n, S[100], D[100];

int main()

{

fin >> n;

int k = 1;

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

for (i = 2;i <= n;i = i + 2)

{

S[k] = i;

if (i < n) D[k] = i + 1;

k++;

}

for (i = 1;i <= n;i++) fout << S[i] << " ";

fout << endl;

for (i = 1;i <= n;i++) fout << D[i] << " ";

fin.close();

fout.close();

return 0;

}