**Seminar 2 Programare SAS**

**Procesare iterativă şi condiţională. Combinarea seturilor de date.**

**Crearea de subseturi. Funcţii SAS.**

**1. Operatori în SAS**

*Operatori de comparaţie*

SAS permite utilizarea operatorilor de comparaţie cunoscuţi, într-o formă specifică. În tabelul de mai jos putem observa sintaxa SAS pentru fiecare operator.

|  |  |  |
| --- | --- | --- |
| **Operator** | **Simbol** | **Cod SAS (mnemonica)** |
| Egal cu | = | EQ |
| Diferit de | ^= or ~= or ¬= | NE |
| Mai mic | < | LT |
| Mai mic sau egal | <= | LE |
| Mai mare | > | GT |
| Mai mare sau egal | >= | GE |
| Intr-o lista |  | IN |

*Operatorul IN (lista-valori)*

Putem folosi acest operator ca alternativă la scrierea a multiple condiţii legate prin OR.

In cadrul listei definite cu IN putem folosi atât valori numerice cât şi caractere, ele fiind separate prin virgulă sau spaţiu.

Exemplu. : Nota IN ( ‘A’ ‘B-’ ‘B+’ ‘C’)

Nota IN ( ‘A’ ,‘B-’,‘B+’,‘C’)

Nota IN ( 10 9 8 7)

Nota IN ( 10,9,8,7)

Valoare IN (10,15,20:30)

*Operatorii logici AND, OR, NOT*

Aceştia sunt utilizaţi în diferite combinaţii pentru a forma expresii complexe. În expresii, în lipsa parantezelor, operaţiile sunt executate in ordinea precedenţei lor, astfel:

1. NOT
2. AND
3. OR

De exemplu:

**if X and Y or Z;**

este echivalentă cu

**if (X and Y) or Z;**

Pentru ca operatorul OR să aibă precedenţă înaintea lui AND trebuie utilizate paranteze, astfel:

**if X and (Y or Z);**

**if X and not y or z;**

este echivalentă cu

**if (X and (not y)) or z;**

**2. Instrucţiunea WHERE**

Dacă se citesc date dintr-un set de date SAS, se poate utiliza instrucţiunea WHERE pentru crearea unui subset de date. De asemenea, se poate utiliza această instrucţiune în cadrul unei proceduri SAS, pentru a crea subseturi ale datelor procesate. Crearea de subseturi poate fi realizată şi cu instrucţiunea IF, dar WHERE oferă o serie de avantaje cum ar fi un număr mai mare de operatori ce pot fi folosiţi, precum şi posibilitatea ca utilizarea acestei instrucţiuni să fie mai eficientă atunci când se lucrează cu un set de date de intrare indexat. De asemenea, instrucţiunea IF nu este permisă în cadrul unei proceduri SAS.

Operatori ce pot fi utilizaţi împreună cu instrucţiunea WHERE

|  |  |  |
| --- | --- | --- |
| **Operator** | **Descriere** | **Exemplu** |
| IS MISSING | Verifică dacă este o valoare lipsă | where varsta is missing |
| IS NULL | Echivalent cu IS MISSING | where varsta is null |
| BETWEEN AND | Interval închis | where varsta between 20 and 40 |
| CONTAINS | Găseşte un subşir | where nume contains lex |
| LIKE | Compară cu un şablon | Where nume like A\_e% |

***Exerciţiul 1.*** Am colectat o serie de date cu privire la studenţi pe baza următoarelor variabile:

* vârsta,
* sex (M sau F)
* proiect (nota la proiect)
* activitate (nota pentru activitatea de seminar)
* examen (nota la examenul final)

Pentru fişierul de date:

21 M 8 9 8

. F 9 9 9

35 M 8 8 8

48 F . . 7

59 F 9 7 9

15 M 8 . 9

67 F 9 8 9

. M 6 7 6

35 F 7 7 8

49 M 5 5 8

Să se creeze un set de date temporar numit Studenti

**data Studenti;**

**length Sex $ 1;**

**input Varsta Sex Proiect Activitate Examen;**

**datalines;**

**21 M 8 9 8**

**. F 9 9 9**

**35 M 8 8 8**

**48 F . . 7**

**59 F 9 7 9**

**15 M 8 . 9**

**67 F 9 8 9**

**. M 6 7 6**

**35 F 7 7 8**

**49 M 5 5 8**

**;**

**title "Date studenti";**

**proc print data=Studenti noobs;**

**run;**

***Exerciţiul 2.*** Să se afişeze din setul de date creat la exerciţiul anterior, toate persoanele de sex feminin care au la proiect nota 9 sau 10, sau au luat nota 10 la examen:

**title "Exemplu de operatori logici";**

Instrucțiunea WHERE face selecția subsetului de date căutat.

**proc print data=work.studenti;**

**where Sex eq 'F' and**

**(Proiect in (9 10) or**

Se specifică variabilele ce vor fi afișate în raportul rezultat.

**examen eq 10);**

**var Sex Activitate Proiect Examen;**

**run;**

**2. Instrucţiunea IF**

***Sintaxa***

* If conditie THEN actiune;
* IF conditie THEN actiune1

ELSE actiune2;

Atunci când expresia logică este evaluată ca fiind adevărată, se execută instrucţiunile ce urmează cuvântului cheie THEN. Dacă expresia nu este adevărată, programul continuă să proceseze următoarele instrucţiuni, sau ramura ELSE dacă aceasta există.

**Atenţie!** Valorile numerice lipsă sunt tratate logic ca cel mai negativ număr ce poate fi referit, astfel încât atunci când o astfel de valoare este testată într-o expresie IF, programul va evalua rezultatul testării ca fiind TRUE.

Atunci când se doreşte verificarea mai multor condiţii printr-o succesiune de clauze IF este recomandată utilizarea variantei IF...ELSE...IF. Acest lucru duce la o evaluare mai eficientă a programului, deoarece SAS nu mai testează condiţiile ELSE rămase atunci când identifică o ramură IF a cărei condiţie este evaluată ca adevărată.

***Exerciţiul 3:*** Folosind datele prezentate la exerciţiul 1, să se creeze o nouă variabilă care să reprezinte grupe de vârstă.

Această funcție returnează TRUE dacă argumentul lipsește, respectiv FALSE dacă acesta nu lipsește

**data Studenti;**

**length Sex $ 1;**

**input Varsta Sex Proiect Activitate Examen;**

**if Varsta lt 20 and not *missing*(Varsta) then GrupVarsta = 1;**

**else if Varsta ge 20 and Varsta lt 40 then GrupVarsta = 2;**

**else if Varsta ge 40 and Varsta lt 60 then GrupVarsta = 3;**

**else if Varsta ge 60 then GrupVarsta = 4;**

**datalines;**

**21 M 8 9 8**

**. F 9 9 9**

**35 M 8 8 8**

**48 F . . 7**

**59 F 9 7 9**

**15 M 8 . 9**

**67 F 9 8 9**

**. M 6 7 6**

**35 F 7 7 8**

**49 M 5 5 8**

**;**

**title "Date studenti";**

**proc print data=Studenti noobs;**

**run;**

Rezultatul rulării acestui exemplu este un raport ce conține, pe lângă datele introduse, grupa de vârstă a fiecărui student, aşa cum se observă în Figura 1.

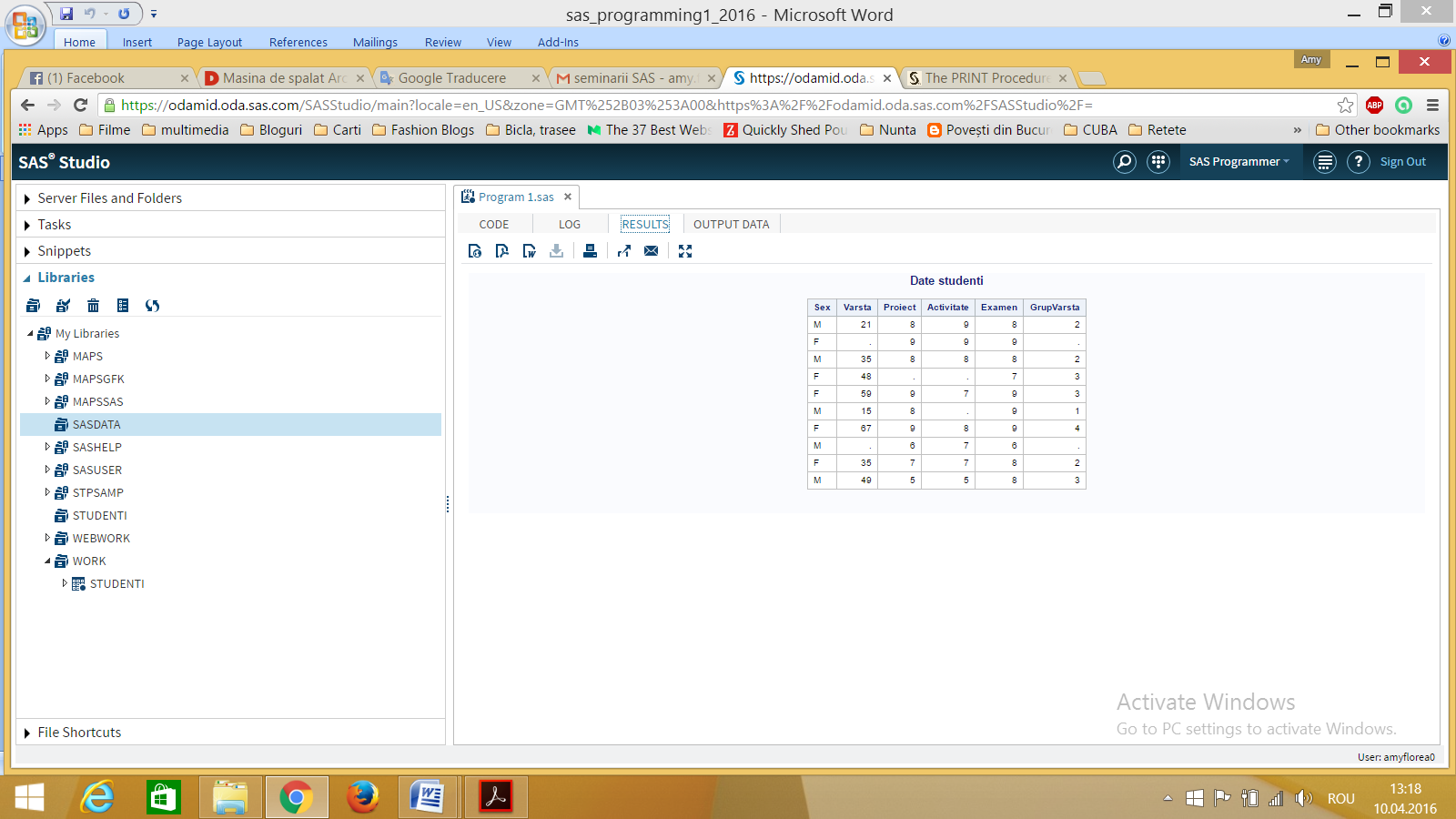


Figura 1. Raportul datelor studenţilor

**Selectarea observaţiilor**

Dacă se doreşte extragerea unui subset de date dintr-un fişier de date sau dintr-un set de date SAS existent, se poate utiliza o formă specială a instrucţiunii IF care permite preluarea exclusiv a acelor rânduri din setul iniţial care îndeplinesc o condiţie specificată.

***Sintaxa:***

IF conditie; /\**Atentie! Lipseşte Then\*/*

unde ‘conditie’ este o expresie SAS.

Dacă condiţia este adevărată, pasul ‘Data’ continuă să proceseze observaţia. Dacă condiţia este falsă nu se mai procesează nimic, controlul revenind la începutul pasului ‘Data’ pentru procesarea următorului rând.

***Exerciţiul 4.*** Folosind setul de date anterior, să se afişeze exclusiv datele studenţilor de sex feminin.

**data Femei;**

**length Sex $ 1;**

**input Varsta Sex Proiect Activitate Examen;**

Dacă valoarea variabilei Sex este ‘F’, atunci se ajunge la finalul pasului Data și are loc un output automat, dacă valoarea este diferită de ‘F’, se revine la inceputul pasului Data.

**if Sex eq 'F';**

**datalines;**

**21 M 8 9 8**

**. F 9 9 9**

**35 M 8 8 8**

**48 F . . 7**

**59 F 9 7 9**

**15 M 8 . 9**

**67 F 9 8 9**

**. M 6 7 6**

**35 F 7 7 8**

**49 M 5 5 8**

**;**

**title "Date studente";**

**proc print data=Femei;**

**run;**

Rezultatul rulării acestui program este prezentat în figura 2.

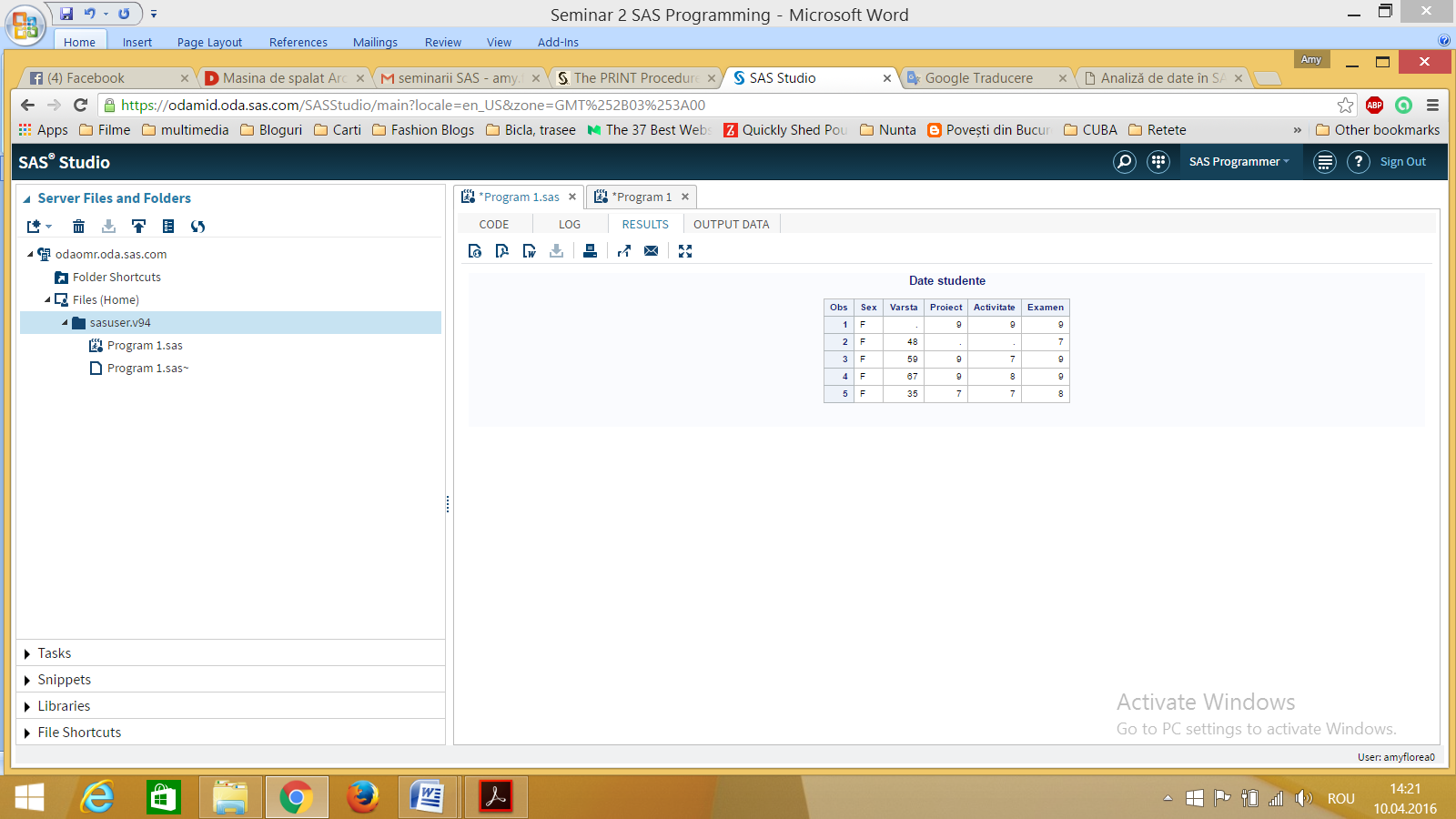


Figura 2. Raport studenţi de sex feminin

**Evitarea trunchierii valorilor la crearea de variabile noi**

La crearea de noi variabile pornind de la expresii condiţionale, SAS stabileşte lăţimea noii variabile în funcţie valoarea primei observaţii din tabelul rezultat. Mai ales pentru variabilele de tip caracter, este posibil să aibă loc trunchieri în cazul în care valorile înregistrărilor următoare trebuie stocate pe o lăţime mai mare decât a primei înregistrări. O soluţie pentru evitarea trunchierilor este stabilirea în avans a lăţimii noii variabile create, asftel încât aceasta să fie acoperitoare pentru toate valorile care se intenţionează a se memora.

***Exercitiul 5.*** Se cunosc datele referitoare la proiectele anuale desfăşurate în cadrul unei companii. Pentru proiectele a căror buget este mai mic sau egal cu 10.000, se va realiza o raportare lunară a rezultatelor, iar pentru cele a căror bugetul depaşeşte această sumă, raportarea va fi bilunară. Se va adăuga o nouă variabilă pentru a evidenția frecvenţa raportării. De asemenea, se va evidenția printr-o variabilă nouă faptul că prima categorie de proiecte va avea ca termen de prezentare a rezultatelor în cadrul consiliului de administare data de 17 decembrie 2018, iar cea de-a doua categorie data de 20 decembrie 2018.

**data proiecte;**

**length Departament $ 9;**

**input CodProiect Departament $ Valoare;**

**datalines;**

**312 Productie 8720**

**313 Achizitii 12570**

**314 Productie 39750**

**315 Desfacere 7380**

**316 Desfacere 18390**

**;**

**run;**

**\*/fara stabilirea latimii frecventei de raportare;**

**data proiecte1;**

**set proiecte;**

**if Valoare le 10000 then**

**do;**

**Raportare="Lunara";**

**Prezentare='17dec2018'd;**

**end;**

**else**

**do;**

**Raportare="Bilunara";**

**Prezentare='20dec2018'd;**

**end;**

**run;**

**\*/cu stabilirea latimii frecventei de raportare;**

**data proiecte2;**

**set proiecte;**

**length Raportare $8;**

**if Valoare le 10000 then**

**do;**

**Raportare="Lunara";**

**Prezentare='17dec2018'd; \*constanta de tip data;**

**end;**

**else**

**do;**

**Raportare="Bilunara";**

**Prezentare='20dec2018'd;**

**end;**

**run;**

**title "Proiecte companie - fara impunerea latimii coloanei Raportare";**

**proc print data=proiecte1 noobs;**

**format Prezentare ddmmyy10.;**

**format Valoare COMMA8.;**

**run;**

**title "Proiecte companie - cu impunerea latimii coloanei Raportare";**

**proc print data=proiecte2 noobs;**

**format Prezentare ddmmyy10.;**

**format Valoare COMMA8.;**

**run;**

Instrucțiunea SET se poate utiliza cu opțiunea KEEP sau DROP pentru a păstra sau a elimina anumite coloane din setul de date.

!Observaţi rezultatele obţinute şi rulaţi procedura de afişare a rezultatelor şi pentru setul de date proiecte2. Rezultatul rulării celor două programe este disponibil în figura 3.

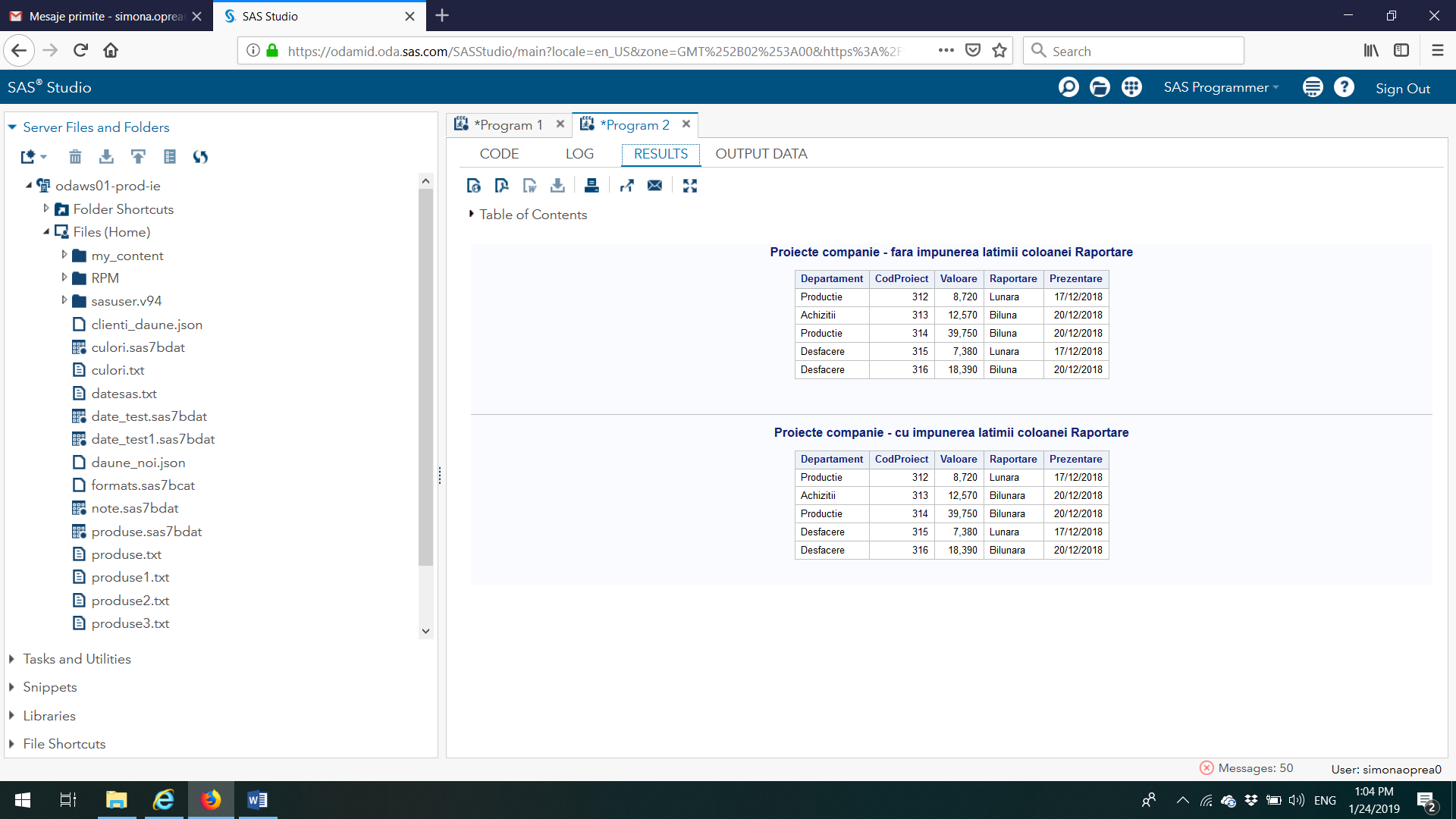


Figura 3. Raport privind proiectele companiei în două variante

**A alege între WHERE sau IF pentru selectarea înregistrărilor**

Ambele intrucţiuni WHERE şi IF se pot folosi pentru a selecta înregistrările care vor fi incluse în noul set de date. Fiecare dintre acestea are însă anumite limite în utilizare, şi anume:

* în secţiunile de proceduri se foloseşte numai instrucţiunea WHERE, nu şi IF;
* dacă WHERE se foloseşte în cadrul secţiunii de date, atunci expresiile sale condiţionale trebuie să refere numai variabile din setul de date de intrare;
* WHERE nu poate folosi în expresii condiţionale bazate pe variabile create în secţiunea de date prin declaraţii de atribuire.

***Exerciţiul 6.*** Să se creeze pornind de la setul de date de la exerciţiul 1, un nou set de date care conţine toate persoanele de sex masculin care au nota finală peste 7, ştiind că nota finală se calculează ca medie ponderată între cele trei note obţinute, cu ponderile de 20%, 30% şi respectiv 50%.

**data Studenti1;  
set Studenti;  
 where Sex eq 'M';  
 NotaFinala=SUM(Proiect\*0.3+Activitate\*0.2+Examen\*0.5);  
 if NotaFinala>7;  
run;  
Title "Lista studentilor de sex masculin cu nota finala peste 7";  
proc print data=Studenti1 noobs;  
run;**

Rezultatul rulării acestui program este prezentată în figura 4.

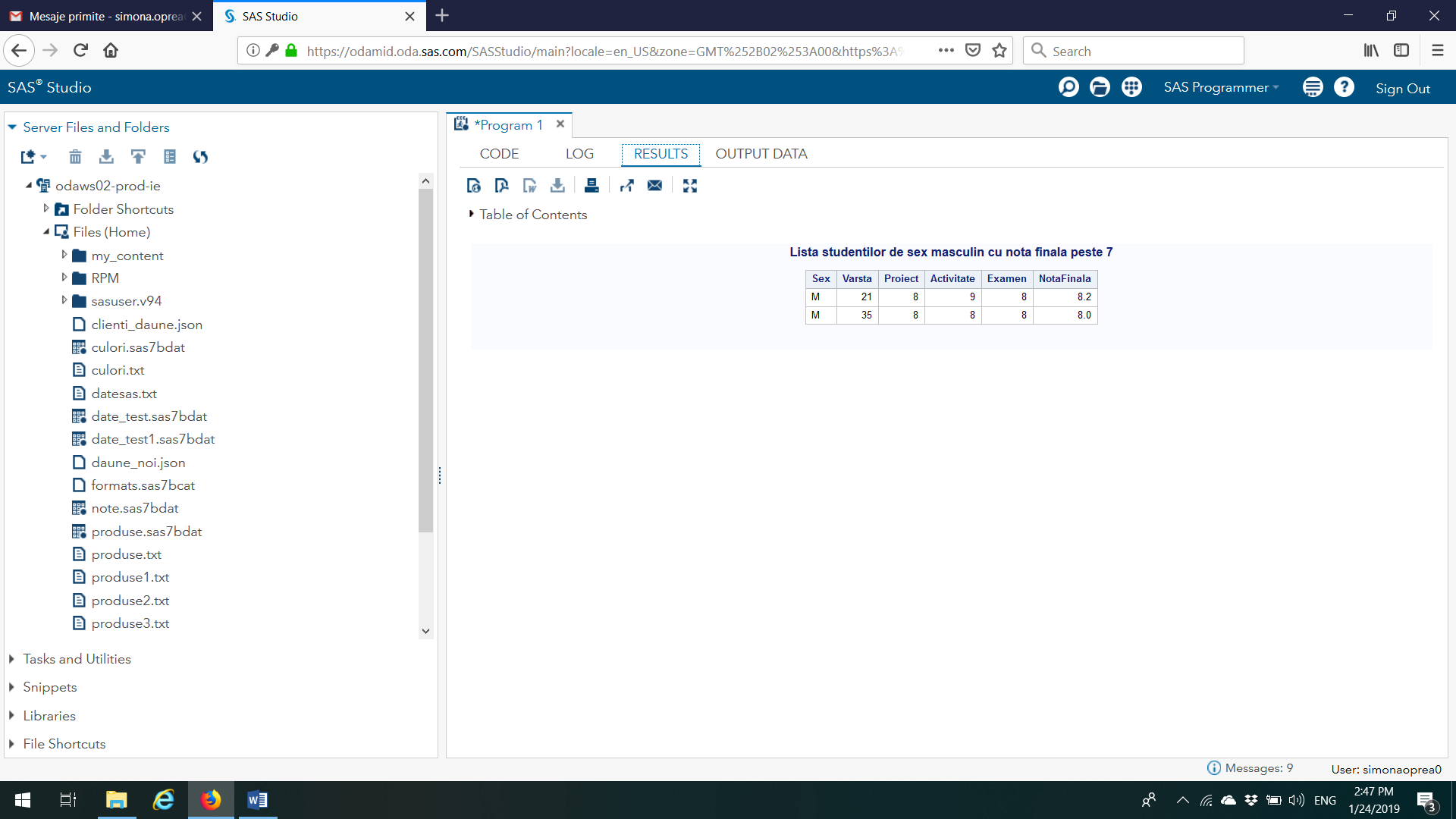


Figura 4. Raport studenţi de sex masculin cu nota finala peste 7

**3. Instrucţiunea SELECT**

Această instrucţiune poate fi utilizată ca o alternativă la o serie de instrucţiuni IF ELSE IF.

***Sintaxa***

SELECT (expresie\_de\_selectie);

WHEN (expresie\_WHEN1) instructiune 1;

WHEN (expresie\_WHEN2) instructiune 2;

…..

OTHERWISE instrucţiune\_implicită;

END;

Expresia de selecţie este comparată cu fiecare expresie din WHEN. Dacă rezultatul comparării este TRUE, atunci se execută instrucţiunea corespunzătoare şi controlul programului trece la finalul instrucțiunii SELECT. Dacă comparaţia nu este adevărată, se evaluează următoarea ramură WHEN. Dacă nici una dintre comparaţiile cu expresiile WHEN nu rezultă ca fiind adevărate, atunci se execută OTHERWISE. *Atentie!* Aceasta poate fi şi null.

***Exemplu:***

Expresie\_de\_selectie

**SELECT (GrupVarsta);**

**WHEN (1) Limit = 110;**

Expresie\_WHEN

**WHEN (2) Limit = 120;**

Instrucţiune

**WHEN (3) Limit = 130;**

NULL pe ramura OTHERWISE

**OTHERWISE ;**

**END;**

**Observatie**! Dacă nu se include ramura OTHERWISE, iar toate comparaţiile anterioare sunt false, execuţia programului se încheie.

Dacă nu este precizată o expresie de selecţie, fiecare ramură WHEN este evaluată pentru a vedea dacă expresie\_WHEN este adevărată sau falsă.

***Exercitiul 7.*** Rezolvaţi cerinţa de la exerciţiul 1 folosind o expresie SELECT.

**data Studenti2;**

**length Sex $ 1;**

**input Varsta Sex Proiect Activitate Examen;**

**SELECT;**

**WHEN (missing(Varsta)) GrupVarsta= . ;**

**WHEN (Varsta lt 20) GrupVarsta=1;**

**WHEN (Varsta lt 40) GrupVarsta=2;**

**WHEN (Varsta lt 60) GrupVarsta=3;**

**OTHERWISE GrupVarsta=4;**

**END;**

**datalines;**

**21 M 8 9 8**

**. F 9 9 9**

**35 M 8 8 8**

**48 F . . 7**

**59 F 9 7 9**

**15 M 8 . 9**

**67 F 9 8 9**

**. M 6 7 6**

**35 F 7 7 8**

**49 M 5 5 8**

**;**

**title "Date studenti introduse cu SELECT";**

**proc print data=Studenti2;**

**run;**

Rezultatul rulării este ilustrat în figura 5.

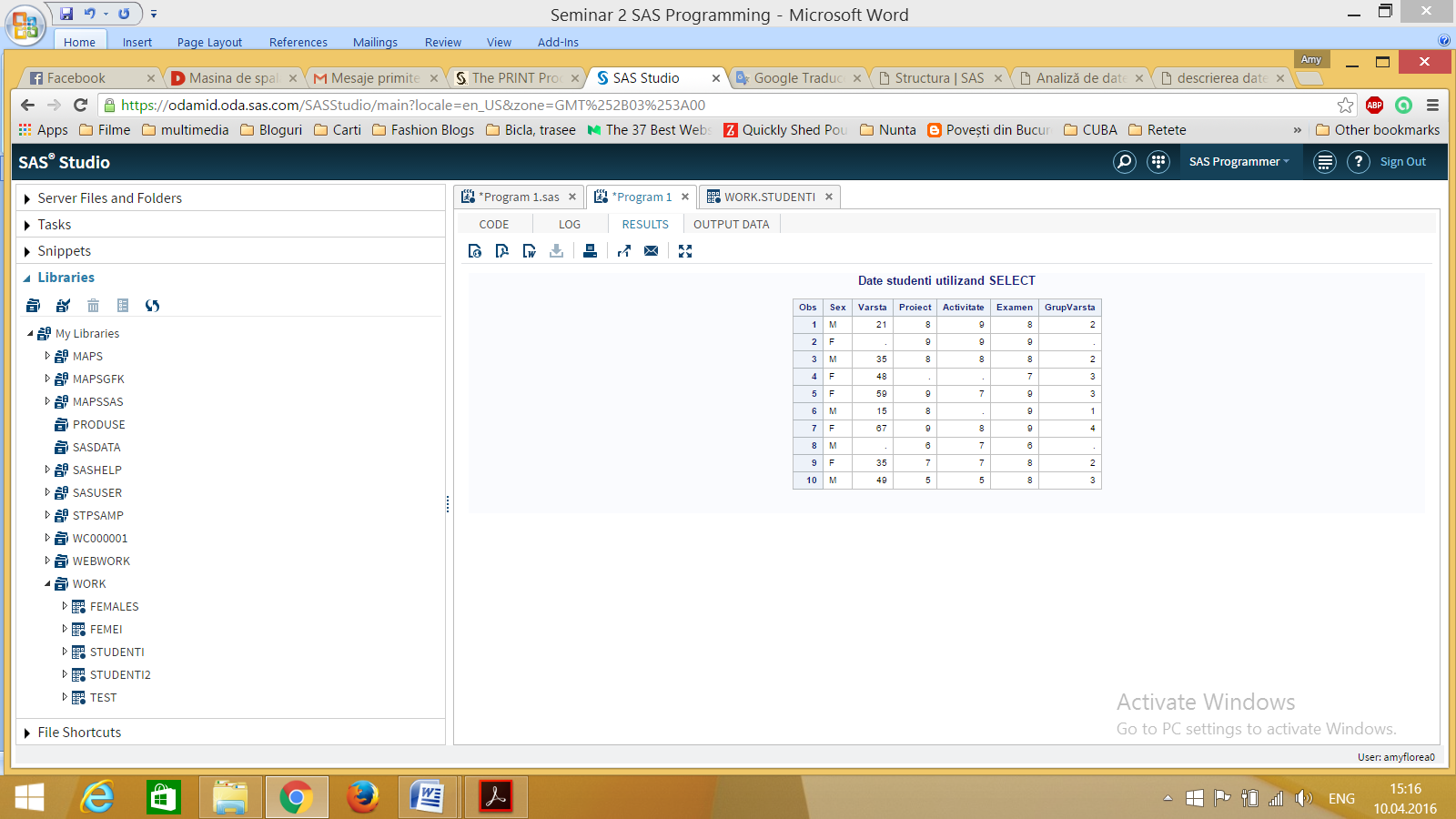


Figura 5. Raport studenţi realizat cu SELECT

**4. Structuri repetitive**

Structurile repetitive permit execuţia unui set de instrucţiuni în mod repetat, cât timp este îndeplinită o anumită condiţie sau până se atinge un număr prestabilit de iteraţii. În SAS putem realiza acest lucru folosind grupurile DO, bucla DO sau instrucţiunile DO WHILE şi DO UNTIL.

**Grupul DO**

***Sintaxa***

Do

Instrucţiune 1;

Instrucţiune 2;

...

Instrucţiune n;

End;

Instrucţiunile cuprinse în cadrul unei structuri Do poartă numele de “grup Do”. Acest mod de lucru este mai eficient decât utilizarea mai multor instrucţiuni IF.

***Exercitiul 8.***  Folosind datele cunoscute, să se adauge două variable noi, GrupVarsta şi Medie calculate după cum urmează: dacă vârsta este mai mică de 39 de ani, valoarea pt GrupVarsta va fi ‘grup1’ iar Media se va calcula ca media ponderată între proiect (40%) şi Examen (60%). Dacă vârsta este mai mare sau egală cu 39 GrupVarstă=’grup2’, iar media se calculează ca medie aritmetică.

**data Medii;**

**length Sex $ 1**

Datele au fost încarcate pe server într-un fisier text numit ‘dateSAS’

**GrupVarsta $ 13;**

**infile '/home/nume.prenume/datesas.txt' missover;**

**input Varsta Sex Proiect Activitate Examen;**

**if missing(Varsta) then delete;**

**if Varsta le 39 then**

**do;**

**GrupVarsta = 'Grup1';**

**Medie = .4\*Proiect + .6\*Examen;**

**end;**

**else if Varsta gt 39 then**

**do;**

Grup DO

**GrupVarsta = 'Grup2';**

**medie = (Proiect + Examen)/2;**

**end;**

**run;**

**title "Raportul Mediilor Studentilor";**

**proc print data=medii noobs;**

**run;**

În cazul în care se întâlneşte o valoare NULL pentru variabila Vârstă, funcţia MISSING returnează valoarea TRUE. Instrucţiunea DELETE împiedică adăugarea valorii curente la setul de date şi forţează întoarcerea la începutul pasului DATA. Rezultatul rulării programului poate fi observat în Figura 6.

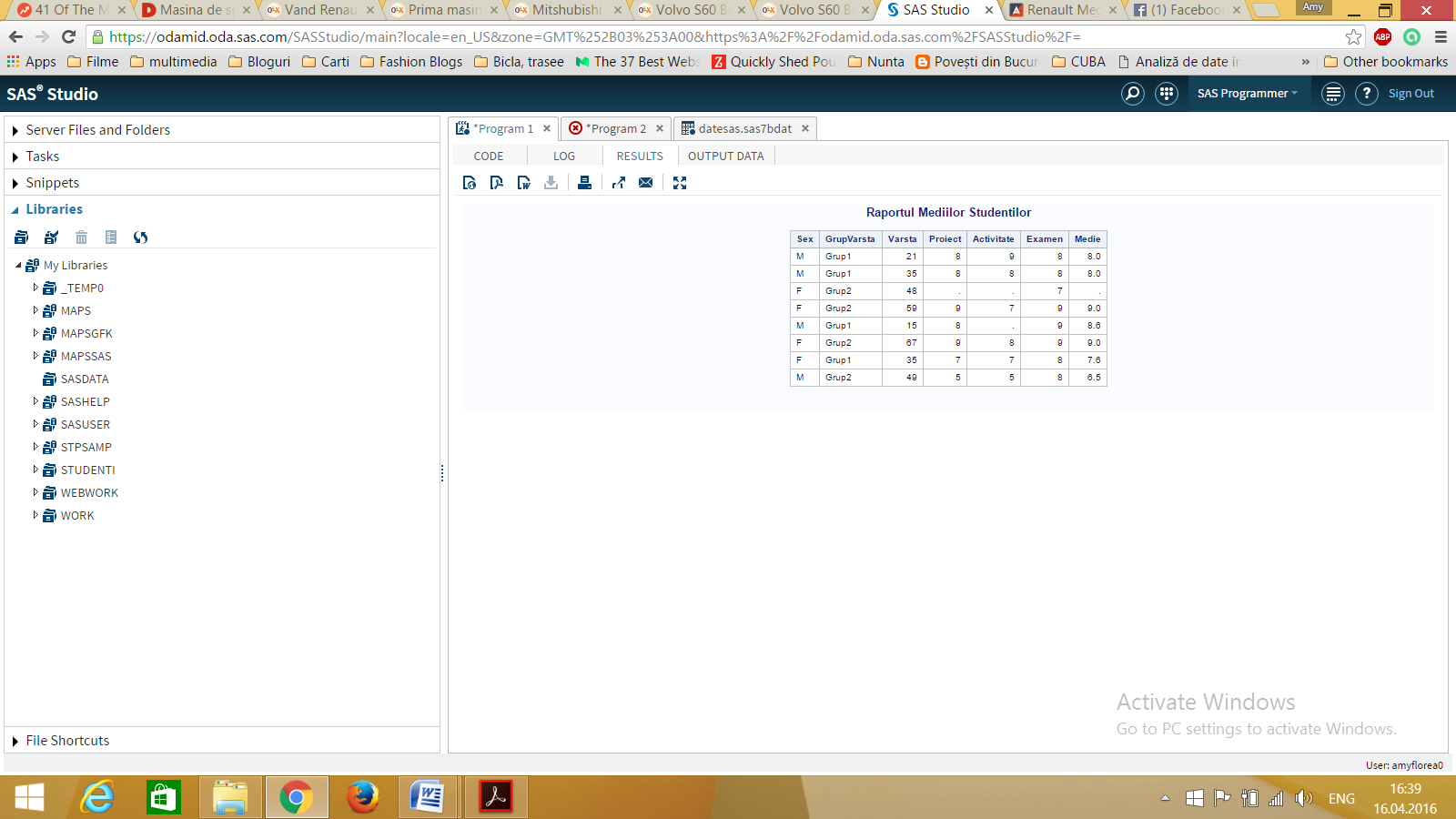


Figura 6. Raportul mediilor studentilor

**SUM Operaţia de adunare**

***Sintaxa***

Variabilă + increment;

Observaţi că nu există semnul = în această instrucţiune. Se realizează următoarele:

* Se reţine variabila;
* Variabila este iniţializată cu 0;
* Valorile lipsă (NULL) sunt ignorate.

***Exerciţiul 9.*** Dacă se doreşte calculul încasărilor dintr-o săptămână, la sfârşitul fiecărei zile a săptămânii, putem folosi SUM.

**data Venit;**

**input**

**Zi : $8.**

**venit : dollar6.;**

**Total + Venit;**

**format Venit Total dollar8.;**

**datalines;**

**Luni $1,000**

**Marti $1,500**

**Miercuri .**

**Joi $2,000**

**Vineri $3,000**

**title "Incasari saptamanale";**

**proc print data=venit noobs;**

**run;**

După citirea celor două variabile cunoscute, cu formatul specificat se calculează valoarea variabilei Total, ca sumă între Totalul existent şi Venitul curent. Dacă valoarea curentă a variabilei venit este NULL, ea va fi ignorată. Rezultatul rulării programului se poate observa în figura 7.

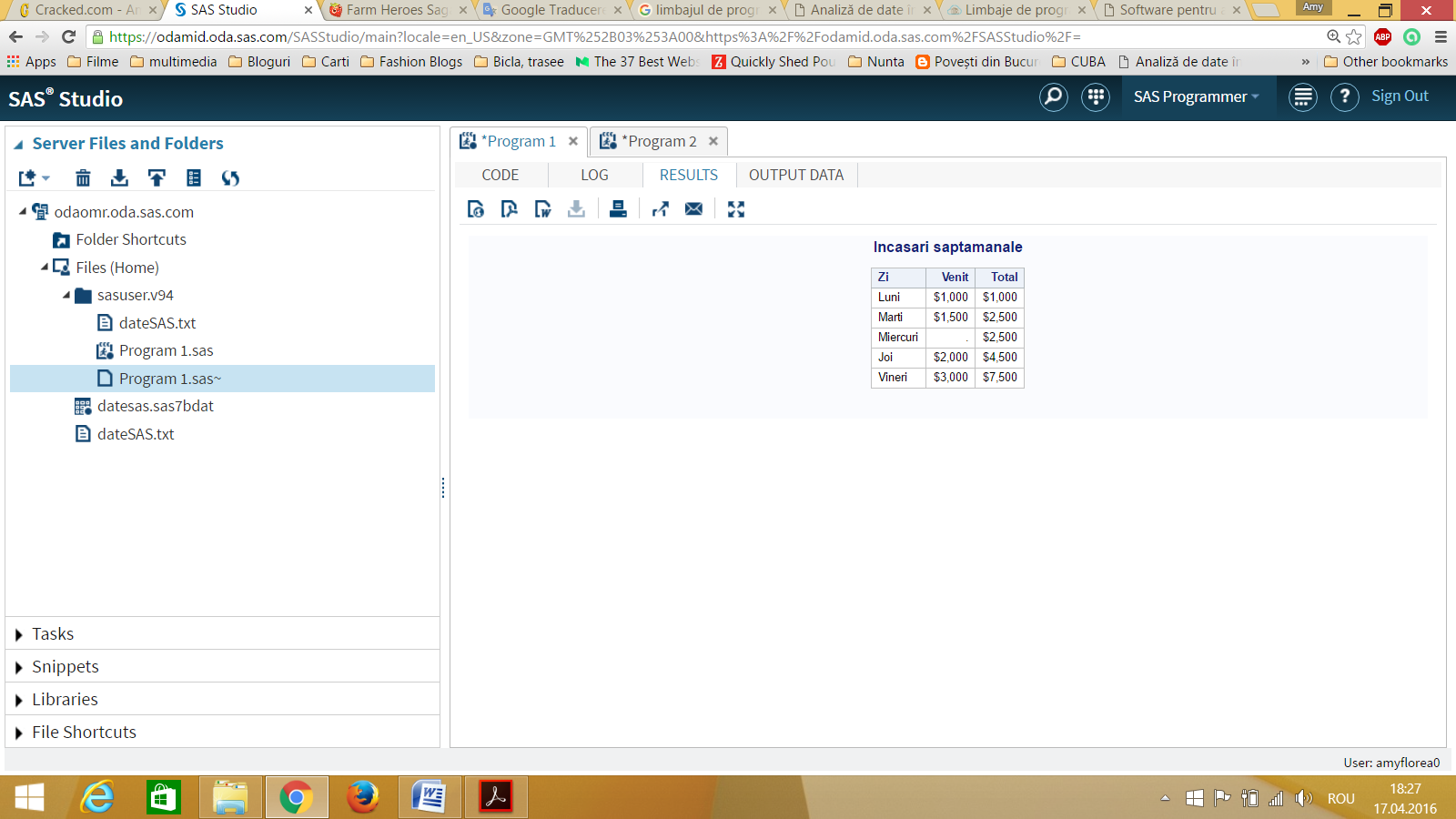


Figura 7. Incasări săptămânale

**Bucla DO (buclă cu un număr cunoscut de iteraţii)**

***Sintaxa***

DO contor=valoare\_start to valoare\_stop;

Set\_instrucţiuni;

END;

O varianta alternativă a sintaxei permite precizarea valorii de incrementare a contorului astfel.

DO contor= *valoare\_start* to *valoare stop* by *valoare incrementare*;

Set\_instrucţiuni;

END;

Atunci când valoarea de incrementare este omisă, valoarea implicită este 1.

De asemenea, contorul poate să realizeze o actualizare descrescătoare, valoarea de incrementare fiind un număr negativ. În această situaţie pentru fiecare iteraţie DO contorul este decrementat cu valoarea specificată Atunci când contorul atinge o valoare mai mică decât valoarea de stop bucla se întrerupe.

***Exerciţiul 10.*** Calculaţi suma de care veţi dispune la finalul a trei ani, plecând de la o investiţie iniţială de 100$, cu o dobândă anuală de 3.75%.

**data Dobanda\_Investitie;**

**Dobanda = .0375;**

**Total = 100;**

**do An = 1 to 3;**

**Total + Dobanda\*Total;**

**output;**

**end;**

**format Total dollar10.2;**

**run;**

**title "Evoluţie investiţie";**

**proc print data=Dobanda\_Investitie noobs;**

**run;**

Comanda **output** introdusă în cadrul buclei DO este o instrucţiune pentru SAS de a scrie o valoare în setul de date de ieşire. Un output are loc de obicei la sfârşitul unui pas de tip DATA, dar în acest caz dorim ca el să aibă loc de fiecare dată când este calculat un nou total. Atunci când includem o comanda **output** în cadrul unui pas DATA (oriunde ar fi inclusă), SAS nu mai execută o scriere automată la finalul pasului DATA. Rezultatul rulării codului de mai sus se observă în figura 8.

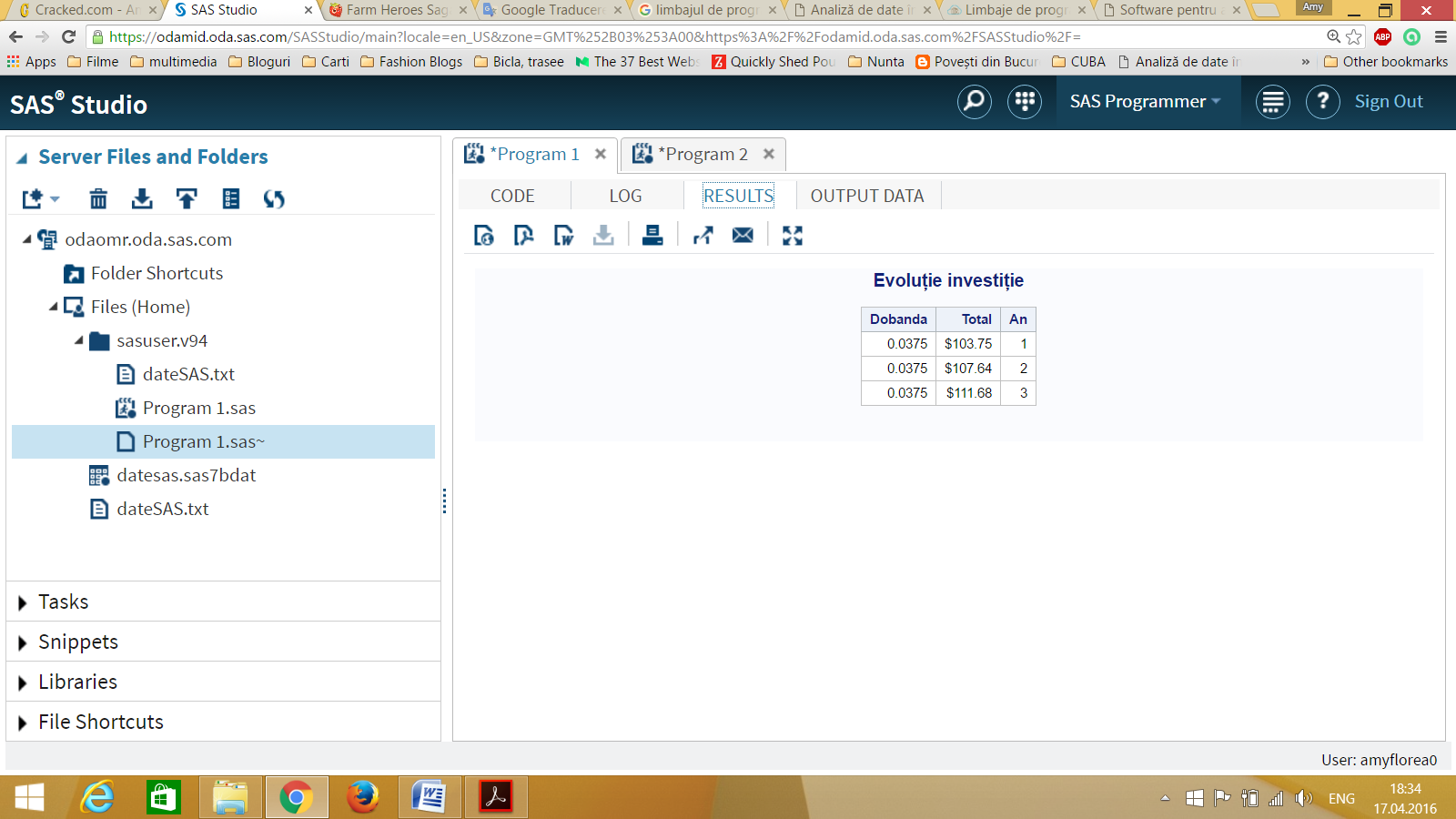


Figura 8. Evoluţie investiţie

**DO WHILE, DO UNTIL**

În loc de a alege o valoare de oprire pentru o buclă iterativă DO, putem să întrerupem iteraţiile atunci când o condiţie este îndeplinită. DO...WHILE este o structură cu verificare anterioară, pe când DO...UNTIL realizează o verificare posterioară.

***Sintaxa***

DO WHILE condiţie

Set\_instrucţiuni;

END;

DO UNTIL (condiţie)

Set\_instrucţiuni;

END;

Observaţie! Atunci când se lucrează cu structura DO UNTIL este foarte important ca condiţia verificată să devină adevărată la un moment dat, pentru a se evita intrarea într-o buclă infinită. O posibilitatea de a evita această situaţie este să combinăm o buclă DO obişnuită cu UNTIL.

***Exerciţiul 11.***  Folosind cele două variante DO, aflaţi câţi ani sunt necesari pentru dublarea investiţiei, dacă suma iniţială este de 100$ şi dobânda este de 3,75%. Rezultatul rulării programului se poate observa în figura 9.

DO WHILE

**data Dublare\_venit;**

**Dobanda = 0.0375;**

**Total = 100;**

**do until (Total ge 200);**

**An + 1;**

**Total = Total + Dobanda\*Total;**

**output;**

**end;**

**format Total dollar10.2;**

**run;**

**title "Dublare venit";**

**proc print data=Dublare\_venit noobs;**

**run;**

DO UNTIL

**data Dublare\_venit;**

**Dobanda = .0375;**

**Total = 100;**

**do while (Total le 200);**

**An + 1;**

**Total = Total + Dobanda\*Total;**

**output;**

**end;**

**format Total dollar10.2;**

**run;**

**proc print data=Dublare\_venit noobs;**

**title "Dublare venit";**

**run;**

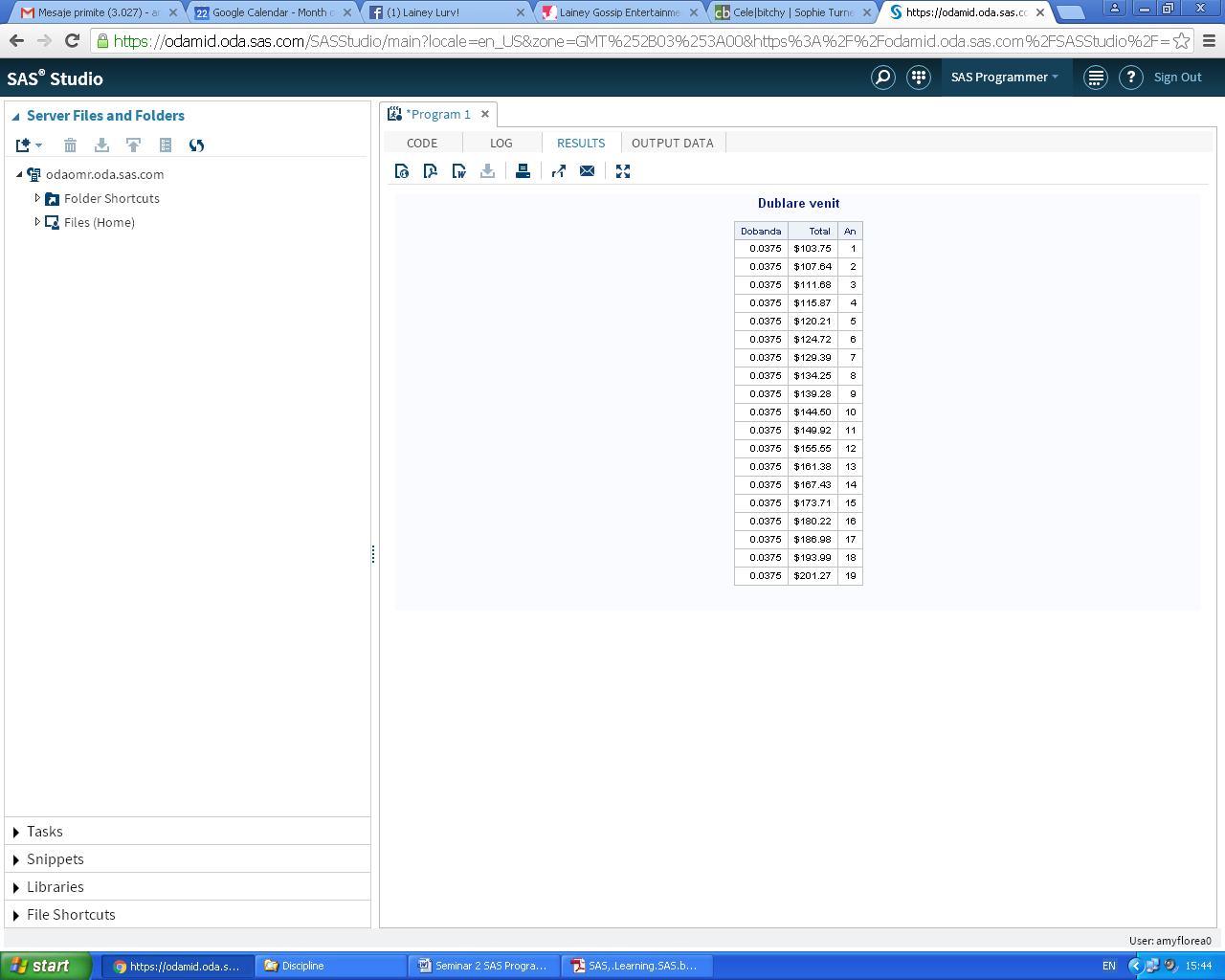
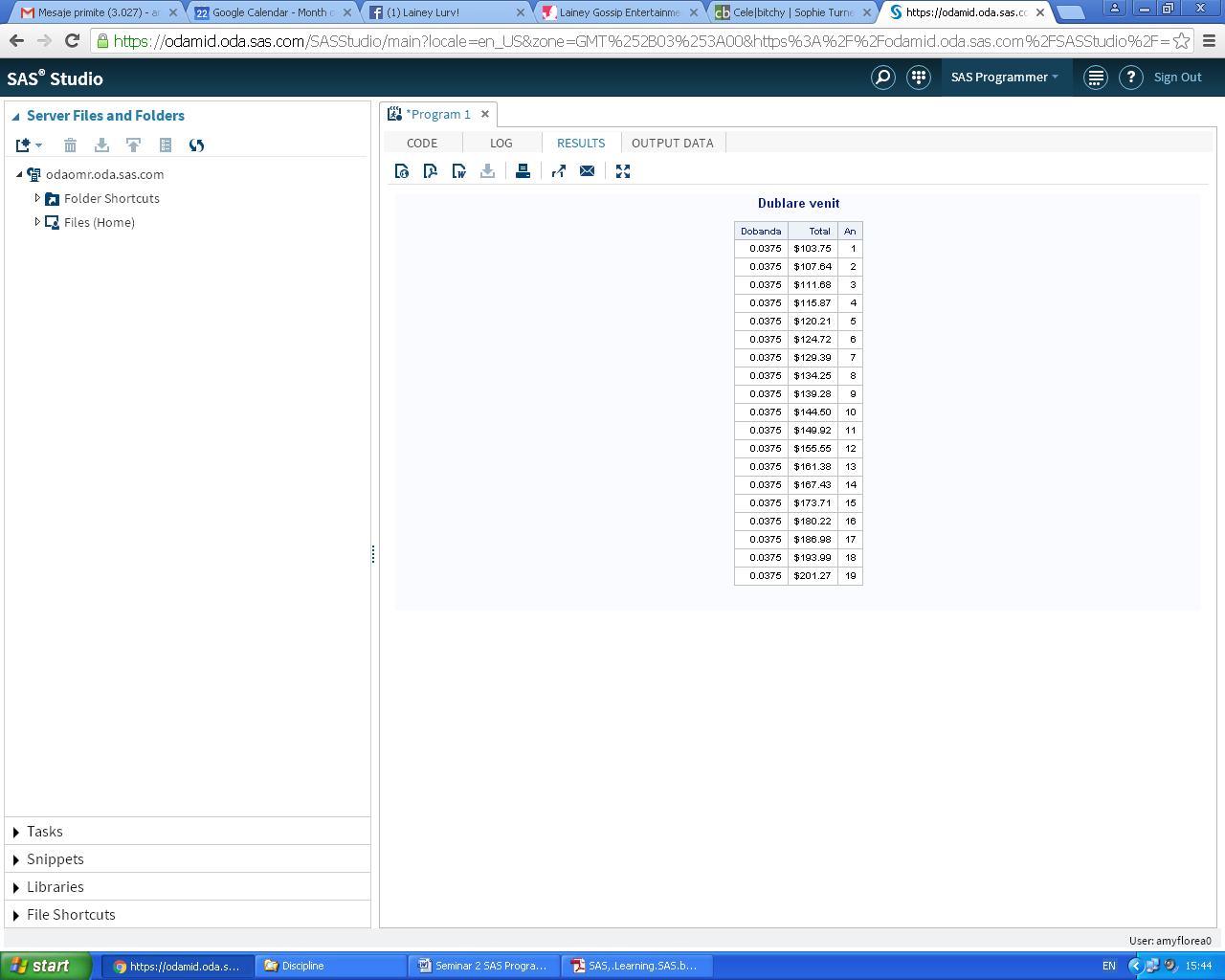
****

Figura 9. Dublare venit

1. **Funcţii SAS**

Funcţiile SAS pot fi folosite pentru a calcula expresii ce stau la baza definirii sau modificării variabilelor SAS. Funcţiile pot lua unul sau mai multe argumente, iar numărul argumentelor este fix sau variabil. Argumentele pot fi constante, variabile sau expresii. Acestea trebuie incluse în paranteze şi separate prin virgulă.

***Exerciţiul 12.***  Se consideră datele referitoare la angajaţii unei companii aeriene. Acestea se regăsesc în setul de date SAS Angajaţi. Se cere să se folosescă funcţii SAS pentru a determina veniturile totale anuale încasate de către fiecare angajat, precum şi luna în care angajaţii primesc prima, cunoscând că:

* toţi angajaţii primesc o primă de 500 de dolari în luna în care au fost angajaţi în companie;
* în funcţie vechimea angajatului, se mai oferă un extra bonus cuprins între 5% si 15% din salariul anual.

Pentru rezolvarea problemei se vor utiliza trei funcţii SAS: SUM(*argument1, argument2,….* ), MONTH(*data\_SAS*) şi YRDIF(data\_inceput, data\_sfarsit, <baza>). O listă completă a funcțiilor predefinite poate fi găsită aici: <http://support.sas.com/documentation/cdl/en/allprodslang/63337/HTML/default/viewer.htm#syntaxByType-function.htm>

Funcţia SUM() returnează suma argumentelor sale şi ignoră valorile lipsă, fiind preferabilă din acest motiv operatorului de adunare care, atunci când primeşte un argument NULL, returnează valoarea NULL.

Funcţia MONTH() extrage luna dintr-o dată calendaristică pe care o are ca argument.

Funcţia YRDIF() returnează diferența în ani dintre două date calendaristice în funcție de convențiile de numărare a zilelor sau poate returna vârsta/vechimea unei persoane. Pentru mai multe detalii despre această funcție se găsesc aici: <http://support.sas.com/documentation/cdl/en/lefunctionsref/63354/HTML/default/viewer.htm#p1pmmr2dtec32an1vbsqmm3abil5.htm>

**libname ad\_data "/home/nume.prenume";  
data work.venituri;  
set ad\_data.angajati;  
prima=500;**

**vechime=YRDIF(DataAngajare, TODAY(), 'ACTUAL');**

**if vechime lt 5 then spor=0.05;**

**else if vechime ge 5 and vechime lt 10 then spor= 0.1;  
else if vechime>=10 then spor =0.15;**

**Venit\_total=sum(salariu\*(1+spor),prima);**

**format Venit\_total DOLLAR10.;  
Luna\_prima=month(DataAngajare);  
run;   
  
title "Venituri totale angajati";  
Proc print data=work.venituri noobs;  
var CodAngajat Nume Prenume Departament Venit\_total Luna\_prima;  
run;**

Rezultatul rulării programului se poate observa în figura 10.

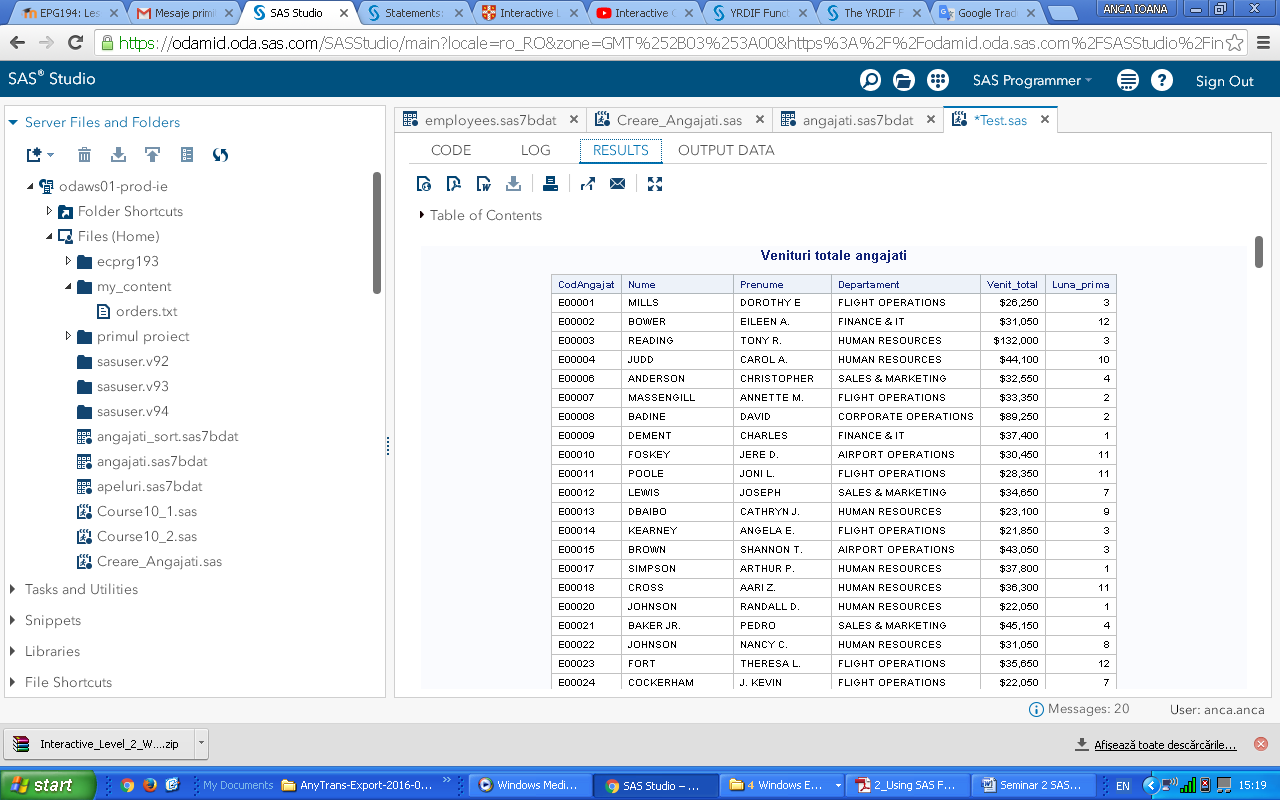


Figura 10. Venituri totale angajati (rezultate parţiale)

1. **Probleme propuse**
2. Utilizând fişierul Excel ‘Vanzari’, creaţi cu SELECT un nou set de date SAS ce va conţine Regiune şi VanzariTotale (utilizând optiunea keep), împreună cu o nouă variabilă numită Ponderi, cu următoarele valori:
   1. 1,5 pentru regiunea Nord;
   2. 1,7 pentru regiunea Sud;
   3. 2 pentru regiunile Est şi Vest.
3. Folosind datele din fişierul Excel ‘Vânzări’, afişaţi toate observaţiile pentru Regiunea Nord şi Cantitatea mai mică de 60. Includeţi în listă orice observaţie unde numele clientului este ***Pet's are Us.***
4. Aveţi o investiţie iniţială de 1000$, cu o rată a dobânzii trimestriale de 4,25%. În câţi ani veţi obţine 30000$? Utilizați DO WHILE sau DO UNTIL pentru rezolvare.
5. Din setul de date Angajati să se afişeze toate persoanele care sunt însoţitori de bord (Codul poziţiei este de tipul “FLTAT1”, “FLTAT2”… “FLTATn). Se vor afişa codul angajatului, numele, prenumele şi departamentul.