Tema 2 - SI

1st Soare Jessica
dept. name of organization (of Aff.)
name of organization (of Aff.)
Brasov, Romania
jessica.soare@student.unitbv.ro

2nd Given Name Surname
dept. name of organization (of Aff.)
name of organization (of Aff.)
City, Country
email address or ORCID

***Abstract*—Acesta este abstractul meu! LaTeX**
***Index Terms*—component, formatting, style, styling, insert**

I. Introduction

Acest document include tema 2.

II. Exercitiul 1

A. Subpunctul 1

Header-ul:
In Header, mi-am definit functiile necesare pentru rezolvarea exercitiului:

```
#ifndef EX1_H
#define EX1_H

#include <vector>
#include <iostream>

using IntArray = std::vector<int>;
IntArray generate(int n);
IntArray bubbleSort(const IntArray &sir);
IntArray merge(const IntArray &sir1,
const IntArray &sir2);
IntArray print(const IntArray &sir);

#endif
```

Fisier sursa:
Aici am scris algoritmii pentru fiecare functie in parte.
Functia **generate** am realizat-o astfel: Am initializat un sir gol de n elemente. Am parcurs sirul, iar pentru fiecare element i-am atribuit o valoare random.
Functia **bubbleSort** am realizat-o astfel: Am salvat sirul intr-o alta variabila. Am parcurs sirul comparand cate doua elemente pe rand, iar daca primul e mai mare ca urmatorul, le interschimb.
Functia **merge** am realizat-o astfel: Am initializat un nou sir gol si i-am alocat dimensiune cat pentru ambele siruri. Cat timp indexul din fiecare sir este mai mic decat marimea acestuia, daca elementul din primul sir e mai mic decat cel din al doilea sir, pun in noul sir elementul din primul, altfel pe cel din al doilea.

Identify applicable funding agency here. If none, delete this.

Functia **print** am realizat-o astfel: Parcurg tot sirul si il afisez element cu element.

```
#include "ex1.h"
#include <cstdlib>
#include <ctime>
#include <iostream>
#include <vector>

using IntArray = std::vector<int>;

IntArray generate(int n) {
    IntArray sir_nou;

    for (int i = 0; i < n; ++i) {
        int numar = std::rand() % 100;
        sir_nou.push_back(numar);
    }

    return sir_nou;
}

IntArray bubbleSort(const IntArray& sir) {
    IntArray sir_sortat = sir;
    int n = sir_sortat.size();

    for (int i = 0; i < n - 1; ++i) {
        for (int j = 0; j < n - 1; ++j) {
            if (sir_sortat[j] > sir_sortat[j + 1])
                int a = sir_sortat[j];
                sir_sortat[j] = sir_sortat[j + 1];
                sir_sortat[j + 1] = a;
            }
        }
    }

    return sir_sortat;
}

IntArray merge(const IntArray& sir1, const IntArray
    IntArray sir_cont;
    sir_cont.reserve(sir1.size() + sir2.size());
```

```
    int i = 0;
    int j = 0;

    while (i < sir1.size() && j < sir2.size()) {
        if (sir1[i] < sir2[j]) {
            sir_cont.push_back(sir1[i]);
            i++;
        } else {
            sir_cont.push_back(sir2[j]);
            j++;
        }
    }

    while (i < sir1.size()) {
        sir_cont.push_back(sir1[i]);
        i++;
    }

    while (j < sir2.size()) {
        sir_cont.push_back(sir2[j]);
        j++;
    }

    return sir_cont;
}

void print(const IntArray& sir) {
    for (int i = 0; i < sir.size(); i++) {
        std::cout << sir[i] << " ";
    }
    std::cout << std::endl;
}
```

B. Subpunctul 2:

In functia main am apelat toate functiile create mai sus:

```
#include "ex1.h"
#include <iostream>
#include <cstdlib>
#include <ctime>
#include <vector>

using IntArray = std::vector<int>;

int main() {
    std::srand(std::time(nullptr));

    std::cout << "Generarea a doua siruri:\n";
    IntArray sir1 = generate(8);
    print(sir1);
    std::cout << "\n";

    IntArray sir2 = generate(8);
    print(sir2);

    IntArray sir1_sortat = bubbleSort(sir1);
    IntArray sir2_sortat = bubbleSort(sir2);

    std::cout << "\n";
    std::cout << "Sortarea sirurilor:\n";
    print(sir1_sortat);
    std::cout << "\n";
    print(sir2_sortat);

    std::cout << "\n";
    std::cout << "Siruri contopite:\n";

    IntArray sir_contopit=merge(sir1_sortat,
    sir2_sortat);
    print(sir_contopit);

    return 0;
}
```

C. Rezultatele exercitiului:

```
jessica@DESKTOP-FIANKP5:~/tema2/ex1$ make compilare
g++ -c ex1.cpp -o ex1.o
g++ -c main.cpp -o main.o
g++ -o exec main.o ex1.o
jessica@DESKTOP-FIANKP5:~/tema2/ex1$ make rulare
./exec
Generarea a doua siruri:
83 86 77 15 93 35 86 92
49 21 62 27 90 59 63 26
Sortarea sirurilor:
15 35 77 83 86 86 92 93
21 26 27 49 59 62 63 90
Siruri contopite:
49 21 62 27 83 86 77 15 90 59 63 26 93 35 86 92 jessica@DESKTO
```

Fig. 1. Exercitiul 1.

III. EXERCITIUL 2

A. Subpunctul 1:

```
compilare:
    g++ -c ex1.cpp -o ex1.o
    g++ -c main.cpp -o main.o
    g++ exec main.o ex1.o
rulare:
    ./exec
```

B. Subpunctul 2:

Atasat gasiti poza **Exercitiul 2** relevanta rezolvarii acestui subpunct.

C. Subpunctul 3:

```
utils:
    g++ -pg -o ex1.o -c ex1.cpp
build: utiles
    g++ -pg -o main.o -c main.cpp
    g++ -pg -o exec main.o ex1.o
    gprof exec gmon.out >> raport.txt
```

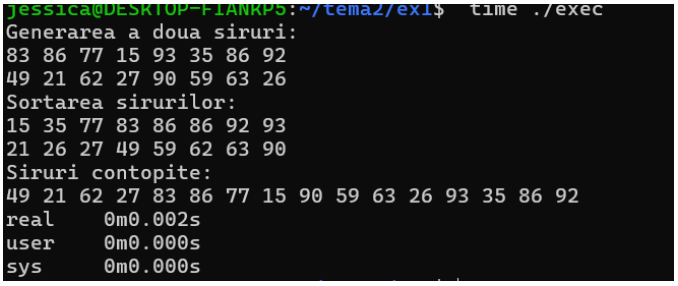

Fig. 2. Exercitiul 2.

D. Subpunctul 4:

FLAGS: -O1

```
utils:
    g++ $(FLAGS) -o ex1.o -c ex1.cpp
build: utiles
    g++ $(FLAGS) -o main.o -c main.cpp
    g++ $(FLAGS) -o exec main.o ex1.o
    gprof exec gmon.out >> raport.txt
profile: FLAGS +=-pg
profile:build
```

IV. Exercitiul 3:

A. Subpunctul 1:

Header-ul:
Am scris intr-un header separat functiile de sortare:

```
#ifndef SORT_H
#define SORT_H

#include <vector>

using IntArray = std::vector<int>;

IntArray bubbleSort(const IntArray &sir);

IntArray selectionSort(const IntArray &sir);

#endif
```

Fisier sursa:
Am realizat functia **selectionSort** astfel: Am parcurs sirul si selectez ca indexul minim primul element, daca urmatorul element e mai mic ca cel cu index minim, indexul minim devine cel al acelui element. La final le interschimb.

```
IntArray sir_s = sir;
int n = sir_s.size();

for(int i=0; i<n-1;i++){

int min_index = i;

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

if(sir_s[j] < sir_s[min_index]){

min_index = j;

}

}

int a = sir_s[i];
sir_s[i] = sir_s[min_index];
sir_s[min_index]= a;

}

return sir_s;
}
```

Makefile-ul:

FLAGS= -O1

```
utils:
        g++ $(FLAGS) -o ex1.o -c ex1.cpp
        g++ $(FLAGS) -o sort.o -c sort.cpp

build: utils
        g++ $(FLAGS) -o main.o -c main.cpp
        g++ $(FLAGS) -o exec main.o ex1.o sort.o
        gprof exec gmon.out >> raport.txt

profile: FLAGS+=-pg
profile: build

clean:
        rm -f *.o

rulare:
        ./exec
timp:
        time ./exec
```

EX1.CPP:

```
while (i < sir1.size()){
sir_cont.push_back(sir1[i]);
i++;

}

while(j < sir2.size())
```

```
{
sir_cont.push_back(sir2[j]);
j++;
}

return sir_cont;
}



void print(const IntArray& sir){

for(int i =0; i<sir.size(); i++){
   std::cout<<sir[i]<<" ";

}
}
```

MAIN.CPP:

```
    IntArray sir1 = generate(8);
print(sir1);
std::cout<<"\n";

IntArray sir2 = generate(8);
print(sir2);

IntArray sir1_sortat = bubbleSort(sir1);
IntArray sir2_sortat = selectionSort(sir2);

std::cout << "\n";
std::cout<< "Sortarea sirurilor:\n";
print(sir1_sortat);
std::cout<<"\n";
print(sir2_sortat);

std::cout<<"\n";
std::cout<<"Siruri contopite:\n";


IntArray sir_contopit = merge(sir1, sir2);
print(sir_contopit);

return 0;
}
```

B. Subpunctul 2:

C. Subpunctul 3:

V. Concluzii

Please number citations consecutively within brackets [**?**]. The sentence punctuation follows the bracket [**?**]. Refer simply to the reference number, as in [**?**]—do not use "Ref. [**?**]" or "reference [**?**]" except at the beginning of a sentence: "Reference [**?**] was the first ..."

```
jessica@DESKTOP-FIANKP5:~/tema2/ex1$  time ./exec
Generarea a doua siruri:
83 86 77 15 93 35 86 92
49 21 62 27 90 59 63 26
Sortarea sirurilor:
15 35 77 83 86 86 92 93
21 26 27 49 59 62 63 90
Siruri contopite:
49 21 62 27 83 86 77 15 90 59 63 26 93 35 86 92
real    0m0.002s
user    0m0.000s
sys     0m0.000s
```

Fig. 3. Exercitiul 3.2.

```
jessica@DESKTOP-FIANKP5:~/tema2/ex1$  time ./exec
Generarea a doua siruri:
83 86 77 15 93 35 86 92
49 21 62 27 90 59 63 26
Sortarea sirurilor:
15 35 77 83 86 86 92 93
21 26 27 49 59 62 63 90
Siruri contopite:
49 21 62 27 83 86 77 15 90 59 63 26 93 35 86 92
real    0m0.002s
user    0m0.000s
sys     0m0.000s
```

Fig. 4. Timpul cu 01.

```
jessica@DESKTOP-FIANKP5:~/tema2/ex1$  time ./exec
Generarea a doua siruri:
83 86 77 15 93 35 86 92
49 21 62 27 90 59 63 26
Sortarea sirurilor:
15 35 77 83 86 86 92 93
21 26 27 49 59 62 63 90
Siruri contopite:
49 21 62 27 83 86 77 15 90 59 63 26 93 35 86 92
real    0m0.002s
user    0m0.000s
sys     0m0.000s
```

Fig. 5. Timpul cu 02.

```
jessica@DESKTOP-FIANKP5:~/tema2/ex1$ time ./exec
Generarea a doua siruri:
83 86 77 15 93 35 86 92
49 21 62 27 90 59 63 26
Sortarea sirurilor:
15 35 77 83 86 86 92 93
21 26 27 49 59 62 63 90
Siruri contopite:
49 21 62 27 83 86 77 15 90 59 63 26 93 35 86 92
real    0m0.002s
user    0m0.002s
sys     0m0.000s
```

Fig. 6. Timpul cu 03.

Number footnotes separately in superscripts. Place the actual footnote at the bottom of the column in which it was cited. Do not put footnotes in the abstract or reference list. Use letters for table footnotes.

Unless there are six authors or more give all authors' names; do not use "et al.". Papers that have not been published, even if they have been submitted for publication, should be cited as "unpublished" [1]. Papers that have been accepted for publication should be cited as "in press" [**?**]. Capitalize only the first word in a paper title, except for proper nouns and element symbols.

For papers published in translation journals, please give the English citation first, followed by the original foreign-language citation [2] Figura 6.

REFERENCES

[1] J. Burns and J. . Gaudiot, "Quantifying the smt layout overhead-does smt pull its weight?" in *Proceedings Sixth International Symposium on High-Performance Computer Architecture. HPCA-6 (Cat. No.PR00550)*, 2000, pp. 109–120.

[2] K. Wu, X. Ma, Z. Zhong, Z. Ming, and C. Tang, "Design of suction nozzle on a new smt silicone rubber keypad," in *2020 21st International Conference on Electronic Packaging Technology (ICEPT)*, 2020, pp. 1–4.