Универзитет „Св. Кирил и Методиј“ - Скопје
ФАКУЛТЕТ ЗА ИНФОРМАТИЧКИ НАУКИ И КОМПЈУТЕРСКО ИНЖЕНЕРСТВО

Structured programming

Exercises 3

Table of Contents

1. Reminder from lectures 1
    1.1. Relational operators 1
    1.2. Logical operators 1
    1.3. Additional operators 2
    1.4. Assignment operator = ... 2
    1.5. Increment and decrement operators 2
    1.6. Compound operators 3
2. Examples 4
    2.1. Variables and assignment 4
    2.2. Casting – `cast` operation 4
    2.3. Casting in another type 4
    2.4. Casting 5
3. Problems 5
    3.1. Problem 1 5
    3.2. Problem 2 6
    3.3. Problem 3 6
    3.4. Problem 4 7
    3.5. Problem 5 8
    3.6. Problem 6 8
    3.7. Problem 7 9
    3.8. Problem 8 9
4. Source code of the examples and problems 11

1. Reminder from lectures

- Operators
  - Arithmetic
  - Relational
  - Logical
- Casting values – **cast** operation

1.1. Relational operators

Are used on all comparable types, and the result is an integer 0 (false) or 1 (true).

Operator	Meaning
<	Less then
<=	Less then or equal
>	Greater then
>=	Greater than or equal
==	Equality
!=	Non equality (different)

1.2. Logical operators

Used in combination with the relational operators to form complex logical expressions, that again return result 0 or 1.

Operator	Operation
&&	Logical **AND**
||	Logical **OR**
!	Negation

Example

```
int a = 5 && 0; // a = 0;
a = 2 && 5;     // a = 1;
a = 0 || 5;     // a = 1;
a = !0;         // a = 1;
a = !5;         // a = 0;
```

1.3. Additional operators

- Assignment operator =
- Incrementation and decrementation operators (++, −−)
  - ++ incrementing (increasing the value for 1)
  - −− decrementing (decreasing the value for 1)
- Using the operators + and – in unary manner

```
X = + Y;
X = - Y;
```

- Compound operators
  - Combination of assignment operator and other operator (+=, -=, *=, /=, %=)

1.4. Assignment operator =

- All expressions have values, even the ones that contain =
- The value of that expression is taken from the right hand side
- So an assignment of this form can be applied:

```
x = (y = 10) * (z = 5);
x = y = z = 20;
```

1.5. Increment and decrement operators

- Increment operator ++ (increasing the value of the operand for 1)
- Decrement operator −−(decreasing the value of the operand for 1)
- Can be applied in **prefix** or **postfix** notation:

1.5.1. Prefix

The value of the variable is incremented **before** the evaluation of the expression it is part of

```
a = ++b;
```

1.5.2. Postfix

The value of the variable is incremented **after** the evaluation of the expression it is part of

```
a = b++;
```

1.6. Compound operators

- Operator +=

Example

```
a += 5; // a = a + 5;
a += b * c; // a = a + b * c;
```

- Operator -=

Example

```
a -= 3; // a = a - 3;
```

- Operator *=

Example

```
a *= 3; // a = a * 3;
```

- Operator /=

Example

```
a /= 3; // a = a / 3;
```

- Operator %=

Example

```
a %= 3; // a = a % 3;
```

2. Examples

2.1. Variables and assignment

```c
#include <stdio.h>
int main () {
    int a;
    float p;
    p = 1.0 / 2.0; /* p = 0.5 */
    a = 5 / 2; /* a = 2 */
    p = 1 / 2 + 1 / 8; /* p = 0; */
    p = 3.5 / 2.8; /* p = 1.25 */
    a = p; /* a = 1 */
    a = a + 1; /* a = 2; */
    return 0;
}
```

2.2. Casting – `cast` operation

Format

```
(data_type) value
```

Example

```c
int i;
double d = 7.28;
i = (int) d;
```

Example

```c
(int) 5.56              // 5.56 in 5
(long) 8.28             // 8.28 in 8L
(double) 2              // 2 in 2.0
(char) 70               // 70 in char with ASCII 70 ('F')
(unsigned short) 3.14   //3.14 in 3 (unsigned short)
```

2.3. Casting in another type

Manipulating the format specification **`%f`** and division operation

```
#include <stdio.h>
int main() {
    int integer1; /* first number entered by the user */
    int integer2; /* second number entered by the user */
    int sum; /* variable for storing the sum */
    float quotient; /* the resut from the division */

    printf("Enter the first integer\n");
    scanf("%d", &integer1);

    printf("Enter the second integer\n");
    scanf("%d", &integer2); /* read integer */

    sum = integer1 + integer2; /* assign the sum with the result from the addition */
    quotient = (float) integer1 / integer2; /* assign the quotient with the result from
the division */

    printf("The sum is %d\n", sum); /* prints the sum */
    printf("Their quotient is %.2f\n", kol); /* prints the quotient */

    return 0;
}
```

2.4. Casting

- In the previous example we used cast operator (converting one type to another):

```
qutient = (float) integer1 / integer2;
```

- Because `integer1` and `integer2` are integers, the result from the integer division would **not be** the expected one. To get a float result one of the values `integer1` or `integer2` should be casted to `float` or `double`.
- The same effect of casting `int` to `double` can be achieved by multiplying the variable with const `double` with value `1.0`

```
kol = 1.0 * integer1 / integer2;
```

3. Problems

3.1. Problem 1

Write a program that reads character from SI and depending if it is lowercase or uppercase will print 1 or 0 accordingly.

Use logical and relational operators for testing the ASCII value of the character.

- Extra: Check if the character is digit

Solution 1

```
#include <stdio.h>

int main() {
    char ch;
    int rez;
    printf("Enter char: ");
    scanf("%c", &ch);
    rez = (ch >= 'a') && (ch <= 'z');
    printf("%d\n", rez);
    return 0;
}
```

Solution of extra

```
rez = (ch >= '0') && (ch <= '9');
```

3.2. Problem 2

Write a program that reads two integers (x, y) and will print the result of (z) the following expression

```
z = x++ + --y + (x<y)
```

What is the value of `z` For `x = 1, y = 2`?

Solution 2

```
#include <stdio.h>

int main() {
    int x, y, z;
    printf("Enter x and y: ");
    scanf("%d%d", &x, &y);
    z = x++ + --y + (x < y);
    printf("z = %d\n", z);
    return 0;
}
```

What will hapen in this situation:

```
z = x++ + --y + x<y;
```

3.3. Problem 3

- Given

```
r = (x<y || y<z++)
```

```
What will be the value of r for x=1, y=2, z=3?
What will be the value of z?
```

- Given:

```
r = (x>y && y<z++)
```

```
What will be the value of r for x=1, y=2, z=3?
What will be the value of z?
```

Solution 3

```c
#include <stdio.h>

int main() {
    int x = 1, y = 2, z = 3, r;
    r = (x < y || y < z++);
    printf("r = %d, z = %d\n", r, z);

    r = (x > y && y < z++);
    printf("r = %d, z = %d\n", r, z);

    return 0;
}
```

Output

```
r=1, z=3

r=0, z=3
```

3.4. Problem 4

- Given:

```c
#include <stdio.h>

int main() {
    int x, y;
    y = scanf("%d", &x);
    printf("y = %d\n", y);
    return 0;
}
```

What will be the value of `y` for `x=5`?

Output

y=1

- Given:

```
#include <stdio.h>

int main() {
    int x, y, z;
    z = scanf("%d%d", &x, &y);
    printf("z = %d\n", z);
    return 0;
}
```

What will be the value z for x=5, y=6?

Output

```
z=2
```

3.5. Problem 5

Write a program where you read from SI price of product, and then will print it's price with calculated with taxes.

The tax is 18% of the price.

Solution 5

```
#include <stdio.h>

int main() {
    float price;
    printf("Enter the product price: ");
    scanf("%f", &price);
    printf("The total price of the product is: %.2f\n", price * 1.18);
    return 0;
}
```

3.6. Problem 6

Write a program where you read from SI price of product, number of installments and interest rate (percents from 0 to 100). The program should output the amount of the installment and total price including the interest.

First compute the total sum, then the installment amount.

Solution 6

```
#include <stdio.h>

int main() {
    float price, interest, amount;
    int installments;
    printf("Enter the product price: ");
    scanf("%f", &price);
    printf("Enter number of installments: ");
    scanf("%d", &installments);
    printf("Enter interest rate: ");
    scanf("%f", &interest);
    float total = price * (1 + interest / 100);
    printf("Installment amount: %.3f\n", total / installments);
    printf("Total payed amount: %.3f\n", total);
    return 0;
}
```

3.7. Problem 7

Read a three digit integer from SI. Then print the most significant and least significant digit.

Example: For the number 795, the program should print:

```
Most significant digit is 7, and least significant digit is 5.
```

Use integer division and modulo operation.

Solution 7

```
#include <stdio.h>
int main() {
    int number;
    printf("Enter number:\n");
    scanf("%d", &number);
    printf("Most significant digit is ");
    printf("%d, and least significant digit is %d.\n", (number / 100),
           (number % 10));
    return 0;
}
```

3.8. Problem 8

Write a program where from the birth date read from SI (in format `ddmmYYYY`) would print the month and day of birth.

Example: For the following input 18091992, the program should print: 18.09

Use integer division and modulo operation.

Try to solve using only the `scanf` function.

Solution 8

```
#include <stdio.h>
int main() {
    long int date;
    printf("Enter birth date in format (ddmmYYYY):\n");
    scanf("%ld", &date);
    int day = date / 1000000;
    int month = (date / 10000) % 100;
    printf("The day and the month of birth are %02d.%02d\n", day, month);
    return 0;
}
```

4. Source code of the examples and problems

https://github.com/finki-mk/SP/

Source code ZIP