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

Structured programming

Exercises 7

Table of Contents

1. Functions

1.1. Reminder from lectures

1.1.1. Defining a function in C

```
return_type name(arguments_list) {
  function_body
}
```

- return_type - type of the return value of the function
- name - name of the function
- arguments_list - list of arguments with names and their types, coma separated
- function_body - the body of the function

1.1.2. Calling (using) function

```
name(arguments_values);
```

- name - name of already defined function
- arguments_values - list of values passed as arguments, coma separated

1.1.3. Example of user defined function

Write a program with a separate function for computing cube of number n^3 за вчитан природен број n.

Example `ex7_1_en.c`

```c
#include <stdio.h>

double cube(int x) {
    return x * x * x;
}

int main() {
    int n;
    scanf("%d", &n);
    double result = cube(n);

    printf("%d^3 = %.2f\n", n, result);
    return 0;
}
```

1.2. Functions from standard library `math.h`

- C has standard library `math.h` with many useful mathematical functions
- To be used, first it needs to be included: `#include <math.h>`
- All functions from the standard library `math.h` accepts arguments of type `double` and return values of same type.

1.3. Most used mathematical functions

Function	Explanation
`sqrt(x)`	square root of x
`exp(x)`	exponential function e^x
`log(x)`	natural logarithm of x (with base e)
`log10(x)`	logarithm of x of base 10
`fabs(x)`	absolute value of x
`ceil(x)`	rounding x to the smallest integer not smaller than x
`floor(x)`	rounding x to the largest integer not larger than x + 1
`pow(x, y)`	x to the power of y
`fmod(x, y)`	residue of x/y as real number
`sin(x)`	sine of x (in radians)
`cos(x)`	cosine of x (in radians)
`tan(x)`	tangent of x (in radians)

1.3.1. Example using function from `math.h`

Write a program that will compute cube n^3 for an integer n.

Example `ex7_2_en.c`

```c
#include <stdio.h>
#include <math.h>

int main() {
    int n;
    scanf("%d", &n);
    double result = pow(n, 3);

    printf("%d^3 = %.2f\n", n, result);
    return 0;
}
```

1.4. Problem 1

Write functions to compute diameter, perimeter and area of circle with radius passed as argument to the functions. Then write a program that for radius read from SI will call the functions to compute the diameter, perimeter and area.

Solution `p7_4_en.c`

```c
#include <stdio.h>
#define PI 3.14

double diameter(double radius);
double perimeter(double radius);
double area(double radius);

int main() {
    double radius, D, L, P;
    scanf("%lf", &radius);

    D = diameter(radius);
    L = perimeter(radius);
    P = area(radius);

    printf("diameter of circle = %.2f\n", D);
    printf("perimeter of circle = %.2f\n", L);
    printf("area of circle = %.2f\n", P);
    return 0;
}

double diameter(double radius) {
    return 2 * radius;
}

double perimeter(double radius) {
    return 2 * radius * PI;
}

double area(double radius) {
    return radius * radius * PI;
}
```

1.5. Problem 2

Write a program that will print all four digit numbers that are divisible with the sum of the numbers formed from the first two and last two digits of the original number. Print how many such numbers exist.

Example:

```
3417 is divisible with 34 + 17
5265 is divisible with 52 + 65
6578 is divisible with 65 + 78
```

Solution `p7_5_en.c`

```c
#include <stdio.h>

int first_two(int n) {
    while(n > 99) {
        n /= 10;
    }
    return n;
}

int last_two(int n) {
    return n % 10;
}

int divisible(int n, int x) {
    return n % x == 0;
}

int main() {
    int i;
    int count = 0;
    for(i = 1000; i <= 9999; ++i) {
        if(divisible(i, first_two(i) + last_two(i))) {
            printf("%d\n", i);
            ++count;
        }
    }
    printf("Total: %d\n", count);
    return 0;
}
```

1.6. Problem 3

Write a program that for a given natural number will compute the difference between that number and the following prime number.

Example: For the number 573, the program should print 577 – 573 = 4

Solution `p7_6_en.c`

```c
#include <stdio.h>

int prime(int n) {
    int i;
    for(i = 2; i * i <= n; ++i) {
        if(n % i == 0) {
            return 0;
        }
    }
    return 1;
}

int first_larger_prime(int n) {
    ++n;
    while(!prime(n)) {
        ++n;
    }
    return n;
}

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

1.7. Problem 4

Write a program that will print all pairs of primes up to 1000 that differentiate between themselves for 2. At the end print the count.

Solution `p7_7_en.c`

```c
#include <stdio.h>

int is_prime(int n) {
    if(n < 4) return 1;
    else {
        if(n % 2 == 0) return 0;
        else {
            int i;
            for(i = 3; i * i <= n; i += 2) {
                if(n % i == 0) {
                    return 0;
                }
            }
        }
    }
    return 1;
}

int sum_digits(int n) {
    int sum = 0;
    while(n != 0) {
        sum += n % 10;
        n /= 10;
    }
    return sum;
}

int main() {
    int i, count = 0;
    for(i = 2; i <= 9999; ++i) {
        if(is_prime(i) && is_prime(sum_digits(i))) {
            printf("%d\t", i);
            ++count;
        }
    }
    printf("\nTotal: %d\n", count);
    return 0;
}
```

1.8. Problem 5

Compute the sum:

1! + (1 + 2)! + (1 + 2 + 3)! + … + (1 + 2 + … + n)!

Use function for computing the sum of the first k natural numbers

Use a function for computing factorial of k

Solution `p7_9_en.c`

```c
#include <stdio.h>

int sum(int n) {
    int i;
    int s = 0;
    for(i = 1; i <= n; ++i) {
        s += i;
    }
    return s;
}

int factorial(int n) {
    int result = 1;
    int i;
    for(i = 1; i <= n; ++i) {
        result *= i;
    }
    return result;
}

int main() {
    int n;
    scanf("%d", &n);
    if(n > 0) {
        int i;
        int result = 0;
        int s;
        for(i = 1; i < n; ++i) {
            s = sum(i);
            result += factorial(s);
            printf("%d! + ", s);
        }
        s = sum(n);
        result += factorial(s);
        printf("%d! = %d\n", s, result);
    } else {
        printf("Invalid value\n");
    }
    return 0;
}
```

2. Source code of the examples and problems

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

Source code ZIP