Министерство цифрового развития, связи и массовых коммуникаций
Российской Федерации
Федеральное государственное бюджетное образовательное
учреждение высшего образования
«Сибирский государственный университет
телекоммуникаций и информатики»
(СибГУТИ)

Кафедра прикладной математики и кибернетики

Современные технологии программирования

Практическая работа №13

«Порядок расчета метрических характеристик ПС»

Выполнил: студент 4 курса
группы ИП-111
Кузьменок Денис Витальевич

Проверил преподаватель:
Зайцев Михаил Георгиевич

Новосибирск, 2024 г.

Цель:

Приобретение практических навыков расчета метрических характеристик ПС: трудоемкости реализации, начальной надежности, структурных параметров на основе постановки задачи.

Задание

1. Написать программу на двух языках программирования для расчета следующих метрических характеристик ПС:
    - структурных параметров ПС:
        - числа уровней иерархии в схеме иерархии логических модулей;
        - количества модулей на каждом уровне иерархии;
        - общего числа модулей в ПС;
    - календарного времени программирования;
    - начальной надежности ПС.
2. На основе постановки задачи рассчитать метрические характеристики программы.
3. Сопоставить расчетные метрические характеристики с характеристиками, полученной в результате реализации программы.
4. С помощью написанной программы рассчитать метрические характеристики для следующих значений **η* 2: 300, 400, 512**. При расчете начального количества ошибок принять **τ = 0.5 Tk**. При расчете календарного времени принять число программистов **n = 5**, число отлаженных в день команд ассемблера **v = 20**.

Ход работы

Программа на языке C# показала следующие результаты:

```
Значение = 300
Число уровней иерархии = 3
Общее число модулей = 42
Eta2k = 2468,64560714876
Nk = 55640,8232178626
Длина программы = 2336914,57515023
Объём ПС = 28672782,7886035
Длина ПС, выраженная в количестве команд ассемблера = 876342,965681337
Календарное время программирования = 23369,1457515023
Время отладки = 11684,5728757512
Начальное количество ошибок = 9557,59426286784
Надежность ПС = 1274,89977538533
==========================================
Значение = 400
Число уровней иерархии = 3
Общее число модулей = 57
Eta2k = 3457,54247590989
Nk = 81290,4973142911
Длина программы = 4633558,34691459
Объём ПС = 59103498,3375069
Длина ПС, выраженная в количестве команд ассемблера = 1737584,38009297
Календарное время программирования = 46335,5834691459
Время отладки = 23167,791734573
Начальное количество ошибок = 19701,1661125023
Надежность ПС = 2342,91838606247
==========================================
Значение = 512
Число уровней иерархии = 4
Общее число модулей = 74
Eta2k = 4608
Nk = 112158,028813292
Длина программы = 8299694,13218363
Объём ПС = 109306349,255769
Длина ПС, выраженная в количестве команд ассемблера = 3112385,29956886
Календарное время программирования = 82996,9413218363
Время отладки = 41498,4706609182
Начальное количество ошибок = 36435,4497519231
Надежность ПС = 3950,9945104115
==========================================
```

Программа на языке Python показала следующие результаты:

```
Значение = 300
Число уровней иерархии = 3
Общее число модулей = 42
Eta2k = 2468.6456071487646
Nk = 55640.823217862635
Длина программы = 2336914.575150231
Объём ПС = 28672782.788603533
Длина ПС, выраженная в количестве команд ассемблера = 876342.9656813366
Календарное время программирования = 23369.14575150231
Время отладки = 11684.572875751155
Начальное количество ошибок = 9557.594262867844
Надежность ПС = 1274.8997753853255
==========================================
Значение = 400
Число уровней иерархии = 3
Общее число модулей = 57
Eta2k = 3457.54247590989
Nk = 81290.49731429106
Длина программы = 4633558.34691459
Объём ПС = 59103498.33750691
Длина ПС, выраженная в количестве команд ассемблера = 1737584.3800929715
Календарное время программирования = 46335.5834691459
Время отладки = 23167.79173457295
Начальное количество ошибок = 19701.166112502306
Надежность ПС = 2342.9183860624707
==========================================
```

```
Значение = 512
Число уровней иерархии = 4
Общее число модулей = 74
Eta2k = 4608.0
Nk = 112158.02881329235
Длина программы = 8299694.132183634
Объём ПС = 109306349.2557693
Длина ПС, выраженная в количестве команд ассемблера = 3112385.2995688627
Календарное время программирования = 82996.94132183633
Время отладки = 41498.47066091817
Начальное количество ошибок = 36435.4497519231
Надежность ПС = 3950.994510411497
==========================================
```

Вывод:

В ходе выполнения лабораторной работы были рассчитаны основные метрические характеристики программного обеспечения (ПС), включая структурные параметры, календарное время программирования и начальную надежность системы.

Листинг программ

Program.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace lab13
{
    class Program
    {
        private static void Main(string[] args)
        {
            List<int> etas = new List<int>()
            {
                300,
                400,
                512
            };

            foreach(int element in etas)
            {
                int i = I(element);
                int k = K(element, i);
                double eta2k = Eta2k(element);
                double nk = Nk(eta2k);
                double n = N(k, nk);
                double v = V(k, nk, eta2k);
                double p = P(n);
                double tk = Tk(n);
                double t = T(tk);
                double b0 = B0(v);
                double tn = Tn(b0, t);

                Console.WriteLine($"Значение = {element}");
                Console.WriteLine($"Число уровней иерархии = {i}");
                Console.WriteLine($"Общее число модулей = {k}");
                Console.WriteLine($"Eta2k = {eta2k}");
                Console.WriteLine($"Nk = {nk}");
                Console.WriteLine($"Длина программы = {n}");
                Console.WriteLine($"Объём ПС = {v}");
                Console.WriteLine($"Длина ПС, выраженная в количестве команд
ассемблера = {p}");
                Console.WriteLine($"Календарное время программирования =
{tk}");
                Console.WriteLine($"Время отладки = {t}");
                Console.WriteLine($"Начальное количество ошибок = {b0}");
                Console.WriteLine($"Надежность ПС = {tn}");

Console.WriteLine("=============================================");
            }
        }

        private static int I(int eta)
        {
            return (int)(Math.Log(eta, 2) / 3 + 1);
```

```
        }

        private static int K(int eta, int i)
        {
            int total = 1;
            for(int j = 1; j < i; j++)
            {
                total += (int)(eta / Math.Pow(8, j));
            }

            return total;
        }

        private static double N(int k, double nk)
        {
            return k * nk;
        }

        private static double Nk(double eta2k)
        {
            return 2 * eta2k * Math.Log(eta2k, 2);
        }

        private static double Eta2k(int eta)
        {
            return eta * Math.Log(eta, 2);
        }

        private static double V(int k, double nk, double eta2k)
        {
            return k * nk * Math.Log((2 * eta2k), 2);
        }

        private static double P(double n)
        {
            return 3 * n / 8;
        }

        private static double Tk(double p)
        {
            return p / (5 * 20);
        }

        private static double T(double tk)
        {
            return tk / 2;
        }

        private static double B0(double v)
        {
            return v / 3000;
        }

        private static double Tn(double b0, double t)
        {
            return t / Math.Log(b0);
        }
    }
}
```

Main.py

```
import math

def i(eta):
    return int(math.log2(eta) / 3 + 1)

def k(eta, i):
    total = 1
    for j in range(1, i):
        total += int(eta / (8**j))
    return total

def n(k, nk):
    return k * nk

def nk(eta2k):
    return 2 * eta2k * math.log2(eta2k)

def eta2k(eta):
    return eta * math.log2(eta)

def v(k, nk, eta2k):
    return k * nk * math.log2(2 * eta2k)

def p(n):
    return 3 * n / 8

def tk(n):
    return n / (5 * 20)

def t(tk):
    return tk / 2

def b0(v):
    return v / 3000

def tn(b0, t):
    return t / math.log(b0)


etas = [300, 400, 512]

for element in etas:
    i_val = i(element)
    k_val = k(element, i_val)
    eta2k_val = eta2k(element)
    nk_val = nk(eta2k_val)
    n_val = n(k_val, nk_val)
    v_val = v(k_val, nk_val, eta2k_val)
    p_val = p(n_val)
    tk_val = tk(n_val)
    t_val = t(tk_val)
    b0_val = b0(v_val)
    tn_val = tn(b0_val, t_val)

    print(f"Значение = {element}")
    print(f"Число уровней иерархии = {i_val}")
    print(f"Общее число модулей = {k_val}")
```

```
print(f"Eta2k = {eta2k_val}")
print(f"Nk = {nk_val}")
print(f"Длина программы = {n_val}")
print(f"Объём ПС = {v_val}")
print(f"Длина ПС, выраженная в количестве команд ассемблера = {p_val}")
print(f"Календарное время программирования = {tk_val}")
print(f"Время отладки = {t_val}")
print(f"Начальное количество ошибок = {b0_val}")
print(f"Надежность ПС = {tn_val}")
print("=============================================")
```