

Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное учреждение
высшего образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт цифрового развития
Кафедра инфокоммуникаций

ОТЧЕТ
ПО ЛАБОРАТОРНОЙ РАБОТЕ №11
дисциплины «Алгоритмизация»
Вариант ____

Выполнил:
Иващенко Олег Андреевич
2 курс, группа ИВТ-б-о-22-1,
09.03.02 «Информационные и
вычислительные машины»,
направленность (профиль)
«Программное обеспечение средств
вычислительной техники и
автоматизированных систем»

(подпись)

Руководитель практики:
Доцент кафедры инфокоммуникации
Воронкин Роман Александрович

(подпись)

Отчет защищен с оценкой _____ Дата защиты _____

Ставрополь, 2023 г.

Тема: «Динамическое программирование»

Порядок выполнения работы

Задача 1. Реализация функция вычисления числа Фибоначчи с помощью динамического программирования.

Таблица 1 – Код программы

```
using System;

class Program
{
    static void Main()
    {
        Console.WriteLine($"Введите значение N");
        Console.Write(">>> ");
        int N = int.Parse(Console.ReadLine());
        Console.WriteLine($"Фибоначчи({N}) разными функциями.");
        Console.WriteLine($"Рекурсивная функция с верхним уровнем({N}) = {Fibonacci(N, "TD")}");
        Console.WriteLine($"Итеративная функция с нижним уровнем({N}) = {Fibonacci(N, "BU")}");
        Console.WriteLine($"Улучшенная итеративная функция с нижним уровнем({N}) = {Fibonacci(N, "Improved")}");
        Console.WriteLine("\nДля завершения работы программы нажмите любую клавишу...");
        Console.ReadKey();
    }

    static int Fibonacci(int n, string func = "TD")
    {
        int[] f = new int[n + 1];

        int FibTopDown(int k)
        {
            if (k <= 1) f[k] = k;
            else f[k] = FibTopDown(k - 1) + FibTopDown(k - 2);
            return f[k];
        }

        int FibBottomUp(int k)
        {
            int[] fib = new int[k + 1];
            fib[0] = 0; fib[1] = 1;
            for (int i = 2; i <= k; i++) fib[i] = fib[i - 1] + fib[i - 2];
            return fib[k];
        }

        int FibBottomUpImproved(int k)
        {
            if (k <= 1) return k;

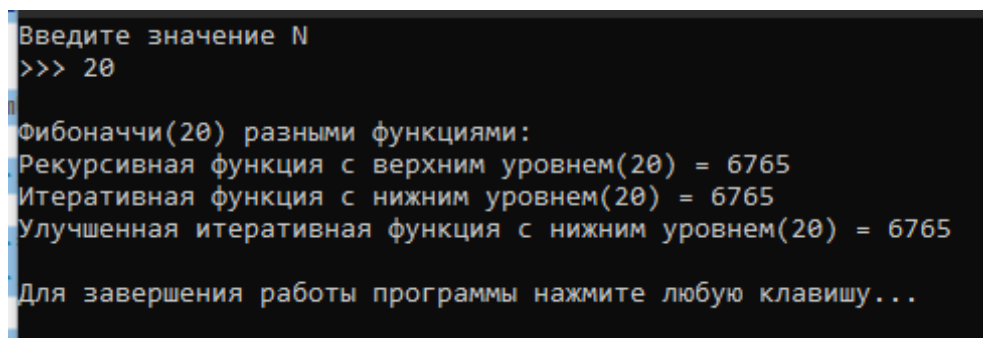
            int prev = 0, curr = 1;
            for (int i = 1; i < k; i++)
            {
                int temp = curr;
                curr = prev + curr;
                prev = temp;
            }
        }
    }
}
```

```

        return curr;
    }

    switch (func)
    {
        case "TD": f = new int[n + 1]; return FibTopDown(n);
        case "BU": return FibBottomUp(n);
        case "Improved": return FibBottomUpImproved(n);
        default:
            Console.WriteLine($"[Ошибка] Неизвестная функция {func}");
            Environment.Exit(1);
            return 0;
    }
}
}

```



```

Введите значение N
>>> 20

Фибоначчи(20) разными функциями:
Рекурсивная функция с верхним уровнем(20) = 6765
Итеративная функция с нижним уровнем(20) = 6765
Улучшенная итеративная функция с нижним уровнем(20) = 6765

Для завершения работы программы нажмите любую клавишу...

```

Рисунок 1 – Результат выполнения программы

Задача 2. Используя динамическое программирование, написать программу для решения задачи о рюкзаке в двух случаях: когда одни и те же предметы можно использовать неограниченное количество раз, и когда каждый предмет может использоваться только один раз.

Таблица 2 – Код программы

```

using System;

class KnapsackProblem
{
    static void Main()
    {
        // Входные данные
        int[] values = { 60, 100, 120 }; // Фиксированные значения объёма
        int[] weights = { 10, 20, 30 }; // Фиксированные значения веса
        Console.WriteLine("Введите значение вместительности:");
        Console.Write(">>> ");
        int capacity = int.Parse(Console.ReadLine()); // Вводимое значение вместительности

        Console.WriteLine("Максимальная стоимость рюкзака (неограниченное количество предметов): " +
            $"{KnapsackUnlimited(values, weights, capacity)}");
        Console.WriteLine("Максимальная стоимость рюкзака (ограниченное количество предметов): " +
            $"{KnapsackLimited(values, weights, capacity)}");
    }
}

```

```

        Console.WriteLine("\nДля завершения работы программы нажмите любую клавишу...");
        Console.ReadKey();
    }

    static int KnapsackUnlimited(int[] values, int[] weights, int capacity)
    {
        int n = values.Length;
        int[] dp = new int[capacity + 1];

        for (int i = 1; i <= capacity; i++)
            for (int j = 0; j < n; j++)
                if (weights[j] <= i) dp[i] = Math.Max(dp[i], dp[i - weights[j]] + values[j]);

        return dp[capacity];
    }

    static int KnapsackLimited(int[] values, int[] weights, int capacity)
    {
        int n = values.Length;
        int[,] dp = new int[n + 1, capacity + 1];

        for (int i = 1; i <= n; i++)
            for (int j = 1; j <= capacity; j++)
                if (weights[i - 1] <= j) dp[i, j] = Math.Max(dp[i - 1, j], dp[i - 1, j - weights[i - 1]] + values[i - 1]);
                else dp[i, j] = dp[i - 1, j];

        return dp[n, capacity];
    }
}

```

```

Введите значение вместительности:
>>> 70
Максимальная стоимость рюкзака (неограниченное количество предметов): 420
Максимальная стоимость рюкзака (ограниченное количество предметов): 280

Для завершения работы программы нажмите любую клавишу...

```

Рисунок 2 – Результат выполнения программы

Задача 3. Используя динамическое программирование, написать программу нахождения наибольшей возрастающей последовательности (НВП).

Таблица 3 – Код программы

```
using System;

class LongestIncreasingSubsequence
{
    static void Main()
    {
        Console.WriteLine("Введите количество элементов: ");
        Console.Write(">>> ");
        int N = int.Parse(Console.ReadLine());
        int[] nums = new int[N];
        for (int i = 0; i < N; i++)
        {
            Console.Write($"[{i}] ");
            nums[i] = int.Parse(Console.ReadLine());
        }

        Console.WriteLine($"Длина наибольшей возрастающей подпоследовательности: {FindLength(nums)}");

        Console.WriteLine("Для завершения работы программы нажмите любую клавишу...");
        Console.ReadKey();
    }

    static int FindLength(int[] nums)
    {
        int n = nums.Length;
        if (n == 0) return 0;

        int[] dp = new int[n];
        dp[0] = 1;
        int maxLength = 1;

        for (int i = 1; i < n; i++)
        {
            dp[i] = 1;

            for (int j = 0; j < i; j++)
                if (nums[i] > nums[j] && dp[i] < dp[j] + 1) dp[i] = dp[j] + 1;

            maxLength = Math.Max(maxLength, dp[i]);
        }
        return maxLength;
    }
}
```

```
Введите количество элементов:  
>>> 10  
[0] 10  
[1] 22  
[2] 33  
[3] 50  
[4] 60  
[5] 80  
[6] 83  
[7] 40  
[8] 57  
[9] 10  
Длина наибольшей возрастающей подпоследовательности: 7  
Для завершения работы программы нажмите любую клавишу...
```

Рисунок 3 – Результат выполнения программы