Platformy programistyczne .Net i Java

# Zadanie 1

Mateusz Trzciński

Politechnika
Wrocławska

| | |
|---|---|
| Wydział: | Wydział Elektroniki Fotoniki i Mikrosystemów |
| Kierunek: | Automatyka i Robotyka |
| Prowadzący: | mgr inż. Michał Jaroszczuk |
| Termin zajęć: | czwartek, $15^{15}$ - $16^{55}$ |
| Temat: | Zadanie 1 |
| Skład grupy: | Mateusz Trzciński 267125 |
| | |
| Data: | 22.03.2025 |

# 1 Link do repozytorium

https://github.com/MTrs00/.Net-Lab1

# 2 Przykładowe fragmenty kodu

```
public Result Solve(int capacity)
{
    Result result = new Result();
    var sorted = items.OrderByDescending(o => o.valueToWeight).ToList();
    // o => o.valueToWeight to wyrażenie lambda, które określa, po jakiej właściwości ma odbywać się sortowanie.


    int storage = 0;

    for (int j = 0; j < itemNumber; ++j)
    {
        if (sorted[j].weight <= capacity - storage)
        {
            result.itemsInside.Add(sorted[j].number);
            result.sumValue += sorted[j].value;
            result.sumWeight += sorted[j].weight;
            storage += sorted[j].weight;
        }
        if (storage == capacity)
            break;
    }
    return result;
}
```

Metoda *Solve*() stosuje algorytm zachłanny, zwraca mieszczące się przedmioty o najwyższym stosunku ceny do wagi. Do sortowania użyto wyrażenia lambda, dzięki któremu można było posorotwać obiekty klasy *Przedmiot* po zmiennej *int valueToWeight*

```
[TestMethod]
Odwołania: 0
public void noMoreFittingItem()
{
    Random random = new Random();
    Result result = new Result();

    Problem problem = new Problem(50, random.Next());
    result = problem.Solve(40);

    bool isInside = false;
    int remainingWeight = 40 - result.sumWeight;

    for(int i = 0; i<50; ++i)
    {
        for(int j=0; j<result.itemsInside.Count; ++j)
        {
            if (problem.items[i].number == result.itemsInside[j])
            {
                isInside = true;
                break;
            }
            else
                isInside = false;
        }
        Assert.IsFalse(problem.items[i].weight < remainingWeight && !isInside);
        Assert.IsFalse(!isInside);
    }

}
```

Testowa metoda *noMoreFittingItem*() rozwiązuje problem (50 przedmiotów i pojemność plecaka 40), następnie oblicza wolne miejsce w plecaku i sprawdza, czy któryś z przedmiotów mogłby się zmieścić, a nie jest włożony.

```
1 odwołanie
private void textBox2_TextChanged(object sender, EventArgs e)
{
    int.TryParse(textBox2.Text, out number);

}

1 odwołanie
private void button1_Click(object sender, EventArgs e)
{
    Problem problem = new Problem(number, seed);
    Result result = new Result();

    result = problem.Solve(50);

    textBox3.Text = problem.ToString();
    textBox4.Text = result.ToString();

}
```

Fragment podstawowego kodu odpowiadającego za pole tekstu i przycisk na GUI. Z powodu braku czasu nie zaimplementowano obsługi błędów