

SPRAWOZDANIE PLATFORMY PROGRAMISTYCZNE .NET I JAVA - LAB1



POLITECHNIKA WROCŁAWSKA

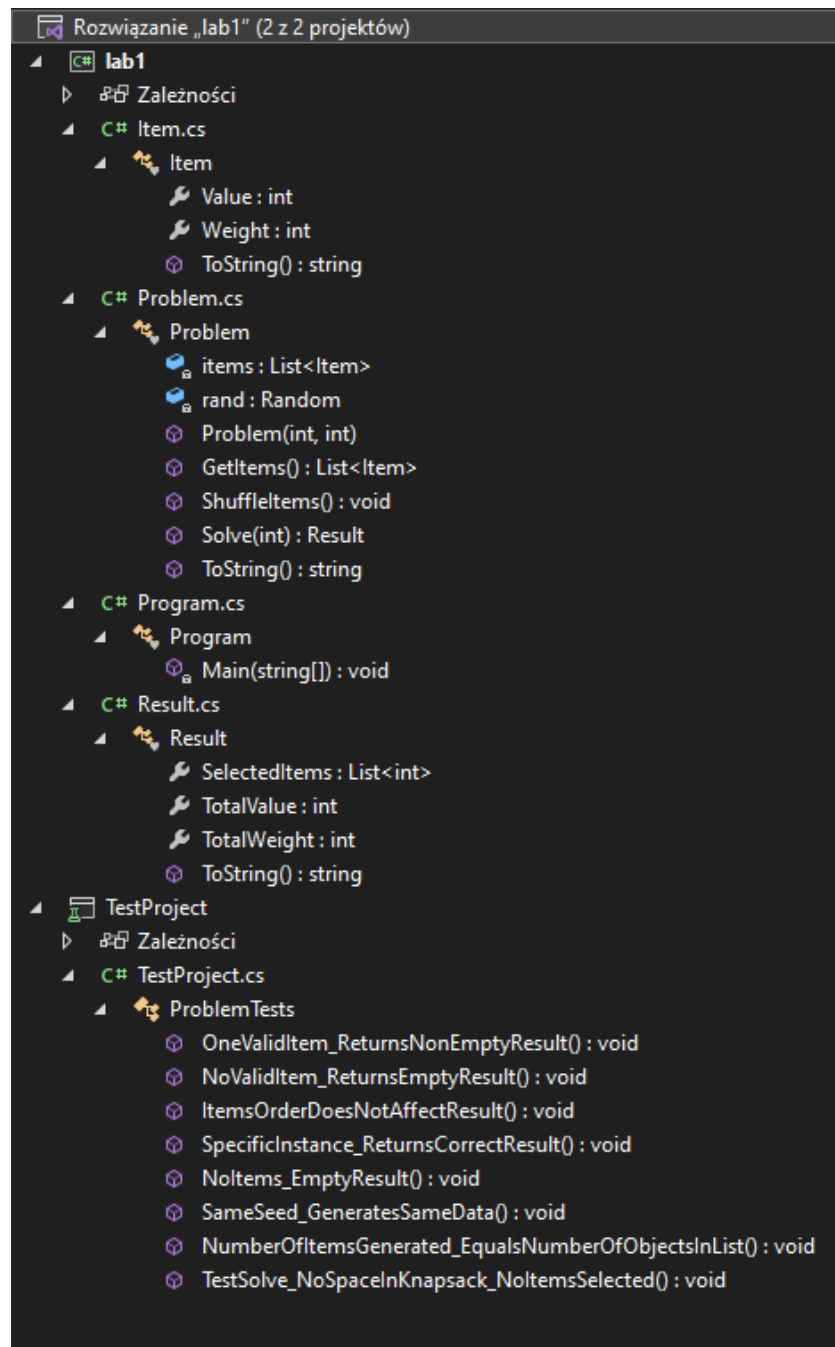
WYDZIAŁ INFORMATYKI I TELEKOMUNIKACJI

INFORMATYCZNE SYSTEMY AUTOMATYKI

MICHAŁ WYRZYKOWSKI

INDEKS 264228

1. Kod C#, drzewo projektu aplikacji konsolowej



```

public Result Solve(int capacity)
{
    items = items.OrderByDescending(x => (double)x.Value / x.Weight).ToList();
    List<int> selectedItems = new List<int>();
    int totalValue = 0;
    int totalWeight = 0;

    foreach (var item in items)
    {
        if (totalWeight + item.Weight <= capacity)
        {
            selectedItems.Add(items.IndexOf(item) + 1);
            totalValue += item.Value;
            totalWeight += item.Weight;
        }
        else
        {
            break;
        }
    }

    return new Result
    {
        SelectedItems = selectedItems,
        TotalValue = totalValue,
        TotalWeight = totalWeight
    };
}

```

Konsola debugowania programu Microsoft Visual Studio

```

Przedmioty:
Item 1: Value: 10, Weight: 10
Item 2: Value: 8, Weight: 9
Item 3: Value: 8, Weight: 1
Item 4: Value: 1, Weight: 2
Item 5: Value: 2, Weight: 7

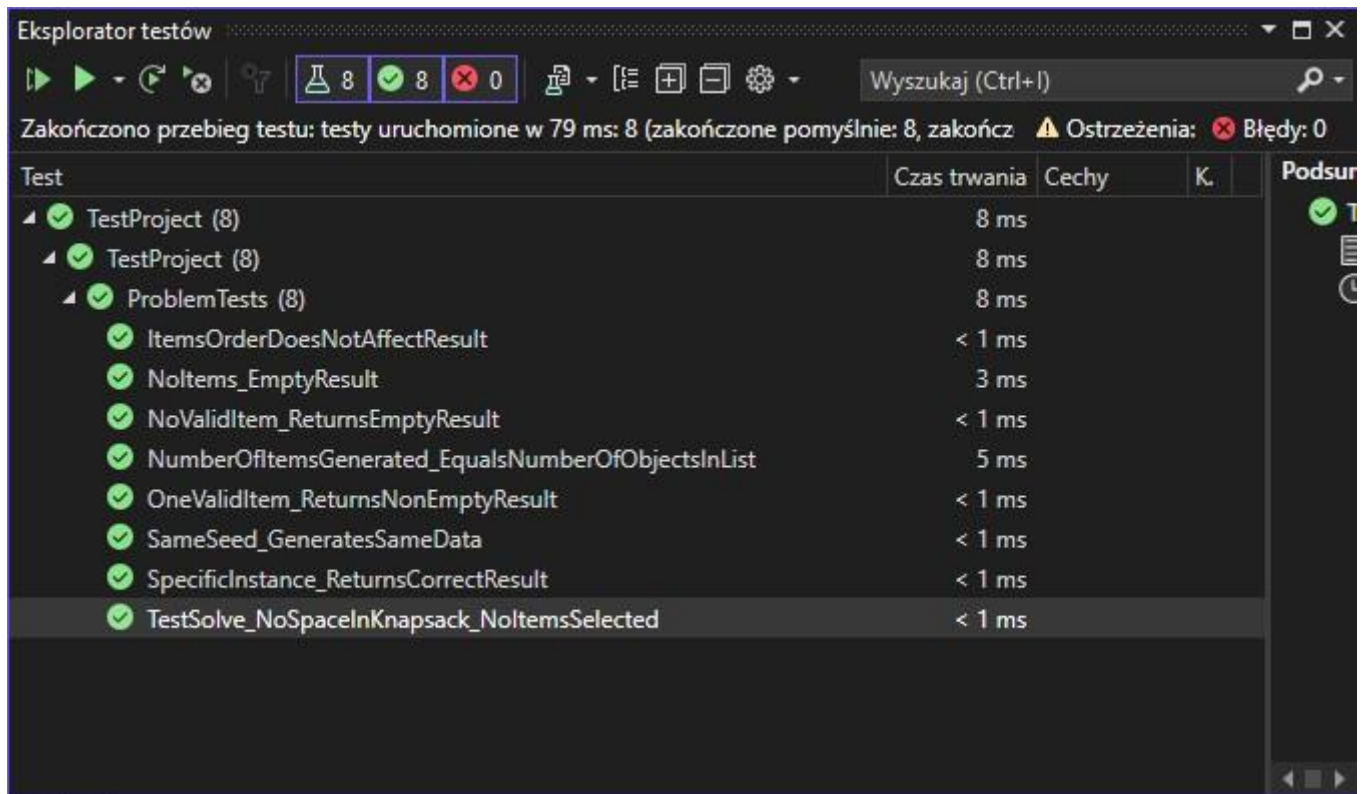
Rozwiązanie dla plecaka o pojemności 20:
Selected items: 1, 2, 3, Total value: 26, Total weight: 20

```

Kod implementuje algorytm rozwiązujący problem plecakowy dla zestawu przedmiotów o określonych wartościach i wagach, przy założeniu, że można wybrać każdy przedmiot tylko raz.

- Sortuje elementy w kolejności malejącej według wartości do wagi stosunku dla każdego elementu.
- Tworzy pustą listę wybranych przedmiotów oraz inicjuje zmienne przechowujące łączną wartość i wagę wybranych przedmiotów.
- Przechodzi przez posortowane elementy:
- Jeśli dodanie aktualnego przedmiotu nie przekroczy pojemności plecaka, to dodaje go do listy wybranych przedmiotów, dodaje jego wartość do łącznej wartości oraz dodaje jego wagę do łącznej wagi.
- Jeśli dodanie przedmiotu spowoduje przekroczenie pojemności plecaka, to przerywa iterację.
- Na koniec zwraca wynik zawierający listę indeksów wybranych przedmiotów, łączną wartość i łączną wagę wybranych przedmiotów.

2. Testy jednostkowe



Eksplorator testów

Zakończono przebieg testu: testy uruchomione w 79 ms: 8 (zakończone pomyślnie: 8, zakończ Ostrzeżenia: 0 Błędy: 0

Test	Czas trwania	Cechy	K.	Podsum
TestProject (8)	8 ms			✓
TestProject (8)	8 ms			
ProblemTests (8)	8 ms			
ItemsOrderDoesNotAffectResult	< 1 ms			
NoItems_EmptyResult	3 ms			
NoValidItem_ReturnsEmptyResult	< 1 ms			
NumberOfItemsGenerated_EqualsNumberOfObjectsInList	5 ms			
OneValidItem_ReturnsNonEmptyResult	< 1 ms			
SameSeed_GeneratesSameData	< 1 ms			
SpecificInstance_ReturnsCorrectResult	< 1 ms			
TestSolve_NoSpaceInKnapsack_NoItemsSelected	< 1 ms			

- OneValidItem_ReturnsNonEmptyResult: Sprawdza, czy rozwiązanie zawiera co najmniej jeden poprawny przedmiot.
- NoValidItem_ReturnsEmptyResult: Sprawdza, czy dla braku dostępnych przedmiotów zwracane jest puste rozwiązanie.
- ItemsOrderDoesNotAffectResult: Sprawdza, czy kolejność przedmiotów nie wpływa na ostateczne rozwiązanie.
- SpecificInstance_ReturnsCorrectResult: Sprawdza, czy dla określonego zestawu przedmiotów i pojemności plecaka zwracane jest poprawne rozwiązanie.
- NoItems_EmptyResult: Sprawdza, czy dla braku przedmiotów zwracane jest puste rozwiązanie.
- SameSeed_GeneratesSameData: Sprawdza, czy dla tego samego ziarna generującego dane, generowane są takie same zestawy przedmiotów.
- NumberOfItemsGenerated_EqualsNumberOfObjectsInList: Sprawdza, czy liczba wygenerowanych przedmiotów jest równa liczbie obiektów na liście.
- NoSpaceInKnapsack_NoItemsSelected: Sprawdza, czy dla braku miejsca w plecaku nie zostają wybrane żadne przedmioty.

3. Aplikacja okienkowa

```
private void button1_Click(object sender, EventArgs e)
{
    int capacity, itemCount, seed;
    if (int.TryParse(textBox1.Text, out capacity) &&
        int.TryParse(textBox2.Text, out itemCount) &&
        int.TryParse(textBox3.Text, out seed))
    {
        Problem problem = new Problem(itemCount, seed);
        Result solution = problem.Solve(capacity);

        // Wyświetlanie rozwiązania
        textBox4.Text = $"Total Value: {solution.TotalValue}" + Environment.NewLine +
            $"Total Weight: {solution.TotalWeight}" + Environment.NewLine +
            $"Selected Items: {string.Join(", ", solution.SelectedItems)}";

        // Wyświetlanie listy przedmiotów
        textBox5.Text = problem.ToString();
    }
    else
    {
        MessageBox.Show("Please enter valid values for capacity, item count, and seed.");
    }
}
```

Form1

Capacity
20

Item count
10

Seed
123

Uruchom

Lista przedmiotów

- Item 1: Value: 8, Weight: 1
- Item 2: Value: 10, Weight: 5
- Item 3: Value: 6, Weight: 3
- Item 4: Value: 10, Weight: 10
- Item 5: Value: 8, Weight: 9
- Item 6: Value: 1, Weight: 2
- Item 7: Value: 2, Weight: 5
- Item 8: Value: 2, Weight: 7
- Item 9: Value: 1, Weight: 9
- Item 10: Value: 1, Weight: 10

Wynik

Total Value: 34
Total Weight: 19
Selected Items: 1, 2, 3, 4