

Spis treści

Programowanie w języku Java	3
Słowo wstępu	3
Informacje o języku	3
Zmienne.....	4
Pierwszy program.....	5
Deklaracja i definicja	7
Klasa String	9
Programowanie funkcyjne	12
Edytor tekstu – MS Word	23
Podstawowe informacje.....	23
Tworzenie dokumentu	27
Formatowanie	30
Akapyty.....	31
Style	35
Obiekty	36
Obiekty graficzne.....	40
Korespondencja seryjna	41
Wydruki	43
Przygotowanie wydruków	44
Arkusz kalkulacyjny MS Excel	48
Podstawowe informacje.....	48
Komórki	52
Zarządzanie arkuszami	57
Reguły i funkcje	59
Adresowanie.....	60
Funkcje	61
Formatowanie	63
Format komórek.....	65
Kopiowanie formatu komórki do innej komórki	66

Wykresy	70
Formatowanie arkusza	76
System kontroli wersji – git	82

Programowanie w języku Java

Słowo wstępu

Programy wykorzystane w poradniku znajdują się na [moim githubie](#).

Aby zimportować sobie wszystkie pliki należy w IntelliJ Idea wybrać kolejno:

File->New->Project from Version Control->GitHub->Git Repository URL->

i wkleić tutaj <https://github.com/idzik78/Zamoy.git>

i od tej pory nie trzeba kopiować osobno plików. Macie wszystkie programy na dysku.

Informacje o języku

Język programowania Java powstał na początku lat 90tych ubiegłego stulecia. Jest on w tej chwili najpopularniejszym językiem w świecie komputerów. Aplikacje napisane przy jego użyciu znajdziemy we współczesnych urządzeniach takich jak smartphone, smartwatch, a nawet pralki, lodówki czy samochody. Aplikacje w Javie obsługują również strony internetowe. Pozwalają nam rozmawiać z innymi ludźmi przez czata.

Poniżej widzimy najprostszy program, który znajduje się w każdej książce o programowaniu. Poniższy program ma za zadanie wyświetlić napis „Witaj Świecie”.

```
public class Zadanie0 {  
    public static void main(String[] args) {  
        System.out.println("Hello World");  
    }  
}
```

Język Java jest językiem programowania wysokiego poziomu. Oznacza to, że nie trzeba już pisać kilkunastu instrukcji, aby wyświetlić napis tak jak to było na początku lat 90tych. Wystarczy napisać instrukcję `println()`. Wadą takiego rozwiązania jest fakt, że Java tak samo jak inne współczesne języki nie ma bezpośredniego dostępu do sprzętu komputera.

Dlaczego Java jest taka popularna? Bo można uruchamiać ją wszędzie tam gdzie jest zainstalowane JVM czyli wirtualna maszyna Javy. To ta maszyna odpowiada za interpretację skompilowanych kodów bajtowych Javy na język zrozumiały dla urządzenia.

Ale od początku. Do edytora wpisujemy kod źródłowy. Pliki źródłowe Javy to pliki o rozszerzeniu `*.java`. Kompilator `javac` tłumaczy te pliki do postaci plików z rozszerzeniem `*.class`. Te ostatnie są interpretowane i wykonywane wszędzie, gdzie jest dostępne JVM, czyli na wspomnianych już wszelkiej maści urządzeniach.

Należy tutaj wspomnieć, że język Java jest oficjalnym językiem programowania na system Android. Dobrze się domyślacie. Brawo. Każdy, kto potrafi programować w Java nie będzie miał problemów z szybką nauką pisania aplikacji na Androida.

Jak już wspomniano, aby uruchamiać programy napisane w Javie trzeba mieć JVM. Należy więc zainstalować JRE czyli Java Runtime Environment. Nie wystarczy to jednak, aby pisać własne programy. Wtedy należy zainstalować JDK, które już w sobie zawiera JRE. Po zainstalowaniu JDK jesteśmy gotowi na.... pisanie programów w notatniku. Czego jeszcze brakuje? Jak myślicie?

Do wygodnej pracy należy zainstalować środowisko programistyczne np.Eclipse, NetBeans lub IntelliJ Idea. Przecież nie chcemy pisać własnych programów w notatniku, kompilować pisząc javac i uruchamiać pisząc java. Ze swojej strony polecam oprogramowanie firmy JetBrains o nazwie IntelliJ Idea. Jest do świetny program do pisania różnorodnych aplikacji konsolowych, okienkowych i webowych w języku Java. Program ten używa inteligentnego podpowiadania, które korzysta z tzw. refleksji, o których będzie mowa później. Na silniku IntelliJ Idea powstało środowisko Android Studio – najlepszy program do programowania aplikacji na Androida.

Zmienne

Języki programowania nie powstałyby, gdyby ich celem było wyświetlanie napisu. Każdy programista przedżej czy później znudzi się pisaniem programów, w których wyświetlanie są napisy. Aby pisać lepsze programy trzeba się nauczyć używać zmiennych. Będziemy wtedy w stanie napisać program, który np. dodaje do siebie dwie liczby. Powstaje pytanie gdzie będziemy te liczby przechowywać?

Wartości liczbowe i znakowe przechowujemy w zmiennych. Każda zmienna ma swój ściśle określony typ. Ten typ obowiązuje od początku do zakończenia programu. Przez cały ten okres typ zmiennej i jej zakres się nie zmienia. Taką cechę nazywamy **statycznym typowanym**

Jakie są typy zmiennych? A jakie dane zwykle będziemy przechowywać w programie. Większość odpowie, że liczby i ciągi znaków. Macie rację. W Javie możemy przechowywać **liczby** (całkowite, rzeczywiste) oraz **ciągi znaków**.

Wiemy już co to jest typ zmiennej. Czym będzie zakres zmiennej? Zastanówmy się czy komputery mogą przechowywać liczby o dowolnej długości? Odpowiedz brzmi NIE. Nic nie jest nieskończone. Nie można w programie przechować nieskończonej liczby. Po pierwsze: przecież ona nie istnieje, ale nawet gdyby istniała, to ile czasu zajmie nam wpisanie jej cyfr do komputera. Zgadzamy się zatem, że komputer jest w stanie przechować liczby o określonej długości. Dobrze. W takim razie jakiej długości lub raczej wielkości są to liczby? Tę wielkość nazywamy rozmiarem zmiennej lub jej **zakresem**.

Java pozwala nam deklarować następujące typy całkowite: byte, short, int, long. Każdy typ jest **typem ze znakiem**. Oznacza to, że w każdym z nich jesteśmy w stanie zapisać

nie tylko liczbę dodatnią i zero, ale także liczbę ujemną. Poniższa tabela przedstawia zakres tych typów.

Nazwa typu	Rozmiar w bajtach	zakres
byte	1	-2^7....2^7-1
short	2	-2^15....2^15-1
int	4	-2^31....2^31-1
long	8	-2^63....2^63-1

I od razu ciekawostka. Podany powyżej rozmiar w bajtach jest taki sam dla każdego urządzenia. Nie jest ważne czy jest to komórka czy samochód. Dlatego Java jest tak ciekawym, prostym i porządnym językiem.

Należy pamiętać, że **domyślnym typem całkowitym** dla operacji arytmetycznych jest typ int. Oznacza to, że typem wyniki dla tych operacji będzie int. Do operacji arytmetycznych zaliczamy między innymi: +,-,/,*,%. W Javie znak / oznacza dzielenie całkowite, a więc należy pamiętać, że dostaniemy resztę z takiego dzielenia.

Wśród **typów rzeczywistych** znajdziemy : float i double. Domyślnym typem jest double i zajmuje one więcej miejsca w pamięci niż float. Oznacza to, że jest on typem bardziej precyzyjnym.

Typ znakowy to typ char. Do tego typu możemy podstawać znaki jako wartości. Należy pamiętać, że typ char może również przechowywać liczby całkowite typu int. Oznaczają one po prostu kod ASCII znaku.

Czy nie wydaje się wam, że brakuje tu jeszcze jednego typu? Brakuje! A gdzie będziemy przechowywać informacje o tym czy np.: liczba a jest większa od zera? Brakujący typ to **typ logiczny boolean**. Może on przyjmować jedną z dwóch możliwych wartości: true albo false.

Wszystkie typy przedstawione powyżej są nazywane typami prostymi lub prymitywnymi, w literaturze **primitive data types**. Są one przechowywane w specjalnej szybkiej pamięci.

Pierwszy program

Omówimy teraz części składowe programu napisanego w języku Java.

```
public class Zadanie0 {  
    public static void main(String[] args) {  
        System.out.println("Hello World");  
    }  
}
```

```
    }  
}
```

Od teraz zaczynamy tłumaczenie kolejnych zagadnień. Od tej pory będzie wyjaśniane tylko to, co jest potrzebne w taki sposób jaki jest niezbędny. Zaawansowani programiści mogą się nie zgadzać do końca z pewnym rzeczami. Zabieg ten jednak jest celowy i prowadzi do stopniowego i lepszego zrozumienia programowania przez początkujących programistów.

Siedziecie wygodnie? To zaczynamy!

Każdy program napisany w Java musi się **zmieścić w klasie** a więc między nawiasami klamrowymi...o tutaj

```
public class Zadanie0 {  
    TUTAJ JESTEM  
}
```

Klasa w której umieszczamy nasz program musi mieć taką samą nazwę jak nazwa pliku. To nie wystarczy, żeby uruchomić program. Aby uruchomić program, musimy **zaimplementować** (napisać) metodę główną w programie. Nazywa się ona main i wygląda następująco:

```
public static void main(String[] args) {  
    TUTAJ WPISUJEMY NASZ KOD  
}
```

Aby nasz program działał wystarczy wpisać kod między klamry.

Wyświetlmy więc napis Hello World.

```
public class Zadanie0 {  
    public static void main(String[] args) {  
        System.out.println("Hello World");  
    }  
}
```

Co robi instrukcja `System.out.println()`? Wyświetla napis podany w cudzysłowie. Teraz aby zobaczyć efekt należy skompilować program i uruchomić.

Wskazówka. Aby nie pisać ostatniej instrukcji można skorzystać z podpowiedzi. Jeśli kurSOR znajduje się w main, napisz sout, poczekaj chwilę i naciśnij enter. Dostaniesz całą instrukcję ;) Wystarczy teraz wpisać Hello Word.

Jeśli nie chce ci się pisać, program znajdziesz [na moim githubie](#). Jeśli skopujesz sobie ten program, to zastanów się w jakim celu tu jesteś? Aby kopiować kod czy uczyć się pisząc

program i wyłapywać błędy. Mam nadzieję, że nie skopujesz już żadnego programu i będziesz sam programował.

Deklaracja i definicja

Zajmiemy się teraz deklarowaniem zmiennych w naszym programie.

Załóżmy, że chcemy napisać program, który dodaje do siebie dwie liczby całkowite. Musimy wtedy zadeklarować dwie liczby całkowite.

```
int firstNumber;
```

Tak wygląda **deklaracja zmiennych całkowitej typu int o nazwie firstNumber**. Ale co ona oznacza? Kompilator ma zarezerwować sobie w pamięci 4 bajty. Zarezerwować to znaczy, że żadna inna zmienna nie będzie tutaj przechowywała swoich wartości poza zmienną firstNumber. Kiedy już zarezerwujemy sobie miejsce w pamięci, to możemy do naszej zmiennej przypisać jakąś wartość.

```
firstNumber=5;
```

Tak wygląda **definicja zmiennej całkowitej tpu int o nazwie firstNumber**. Od teraz w zarezerwowanych komórkach pamięci operacyjnej RAM będzie znajdowała się wartość 5.

Ustaloną wartość można oczywiście zmieniać. Robimy to poprzez nadanie definicji czylej

```
firstNumber=15;
```

Od tej chwili zmienna firstNumber będzie miała wartość 15.

Zadeklarujmy drugą zmienną. Nasz program będzie teraz wyglądał następująco:

```
public class Zadanie1 {
    public static void main(String[] args) {
        int firstNumber=15;
        int secondNumber=13;
    }
}
```

Czego nam jeszcze brakuje? Co mamy zrobić? Przypominam, że powinniśmy obliczyć sumę dwóch liczb. W tym celu potrzebujemy jeszcze jednej zmiennej, która będzie przechowywała nasz wynik. Zróbmy to:

```
int suma=firstNumber+secondNumber;
```

I to wszystko. Trzeba jeszcze wyświetlić wynik...Po wszystkich zmianach nasz program będzie wyglądał następująco:

```
public class Zadanie1 {  
    public static void main(String[] args) {  
        int firstNumber=5;  
        int secondNumber=13;  
        int suma=firstNumber+secondNumber;  
        System.out.println("The sum is "+suma);  
    }  
}
```

Zastanówmy się teraz, czy można zoptymalizować nasz program? Pewnie, że można. Może po prostu dodawanie wykonamy podczas wyświetlania wyniku?

```
System.out.println("The sum is "+firstNumber+secondNumber);
```

Po skompilowaniu i uruchomieniu programu dostajemy

The sum is 513

Czy wiemy dlaczego? Nie wiemy. Dlatego wyjaśnię.

Mamy tutaj pierwszy raz do czynienia z bardzo ważną strukturą danych, którą jest klasa **String**. Tej struktury będziemy używali bardzo często. String reprezentuje ciąg znaków. Nie jest on typem prymitywnym. Świadczy o tym np.: to, że piszemy go wielką.

Co oznacza taki zapis ?

```
"The sum is "+5+13
```

Zwróćmy uwagę, że po lewej stronie stoi String . Bardzo silnie przyciąga on elementy stojące po jego prawej stronie. Tak silnie, że dokleja do siebie kolejne elementy, a więc na początku dostaniemy

```
"The sum is 5"+13
```

I znowu nastąpi doklejenie. Tym razem wynikiem będzie

```
"The sum is 513"
```

I wszystko jasne? Ktoś zada pytanie czy da się dokleić 18 ? Da się tylko, że trzeba zapisać dodawanie w nawiasie okrągłym.

```
System.out.println("The sum is "+(firstNumber+secondNumber));
```

Tutaj widzimy

```
"The sum is "+(firstNumber+secondNumber)
```

Teraz String znowu próbuje dokleić do siebie wyrażenie z prawej strony, ale natrafia na opór. Nawias mówi, aby poczekał chwilę, aż wyrażenie w nawiasie zostanie obliczone. Zostanie wykonane:

```
"The sum is "+(18)
```

Liczba 18 jako wynik może już zostać doklejona. I to cała tajemnica!

Wiemy już więc jak deklarować i definiować typy całkowite. Wiemy również, że trzeba uważać na klasę String, z którym już niedługo ponownie się spotkamy.

Podobnie deklarujemy inne typy całkowite, rzeczywiste, logiczne i znakowe.

Jako uzupełnienie wiedzy polecam filmy mojego autorstwa:

[Wyświetlanie napisu](#)

[Deklaracja i definicja zmiennych typów całkowitych](#)

[Deklaracja i definicja zmiennych typów rzeczywistych](#)

[Deklaracja i definicja znaków](#)

[Deklaracja i definicja typu logicznego](#)

[Dodawanie dwóch liczb](#)

Rada: Nie trzeba pisać osobno deklaracji i definicji. Można jedno i drugie napisać w jednej instrukcji i w jednym wierszu.

```
int firstNumber=15;
```

Klasa String

Nadeszła pora, aby opowiedzieć o wyjątkowej strukturze danych którą jest **String**. Przechowuje ona łańcuchy znaków. Jego deklaracja wygląda następująco:

```
String word;
```

Jak widać nie różni się ona od deklaracji typów prostych. Nie tylko to sprawia, że String jest podobny do prymitywnych typów danych. Dzieje się tak również dlatego, że jego definicja jest także bardzo prosta.

```
word="Napis";
```

Co jak już wiemy możemy zapisać równoważnie jako:

```
String word="Napis";
```

Spróbujmy teraz porównać dwa Stringi. Zdefiniujemy je następująco:

Pierwszy sposób definiowania Stringa

```
String napis1="Java";
String napis2="Java";
if (napis1==napis2) System.out.println("Takie same");
else System.out.println("Różne");
```

Jak myślicie jaki będzie wynik porównania Stringów o takich samych zawartościach? Założę się, że podacie wynik TAKIE SAME. Macie rację chociaż nie wiecie że dzieje się tak przez przypadek. Zaraz dowiecie się dlaczego.

Zdefiniujmy teraz Stringi tak jak na Stringi przystało. String jest klasą a klasy definiuje się z użyciem słowa kluczowego new. Będzie o tym jeszcze mowa na późniejszym etapie.

Drugi sposób definiowania Stringa

```
String napis1=new String("Java");
String napis2=new String("Java");
if (napis1==napis2) System.out.println("Takie same");
else System.out.println("Różne");
```

Jaki będzie wynik? Pewnie znowu powiecie TAKIE SAME. Teraz nie macie racji mimo, że zawartość każdego Stringa jest cały czas identyczna!

Co się tutaj dzieje?

Przeanalizujmy jeszcze raz pierwszy przypadek. Przecież napisaliśmy ==, Stringi są takie same, instrukcja IF jest ok? Coś jest nie tak? Okazuje się, że wszystko działa jak powinno. A przyczyną jest niewłaściwe użycie znaku == oraz sposób definicji Stringów.

Po pierwsze należy zapamiętać, że znaku porównania == (podwójny znak równości) używamy jedynie w przypadku porównywania typów prostych, a String nim nie jest! Co wobec tego jest porównywane? Porównywane są zmienne napis1 i napis 2 a nie Stringi! Napis1 i napis2 to odwołania do pamięci w której jest obiekt String zapisany! To dlaczego nie wyszło RÓŻNE!! A dlatego, że w tym przypadku zmienna napis1 i napis2 wskazuje na to samo miejsce pamięci. Dzieje się tak dlatego, że

1. String napis1 dostaje obszar w pamięci i zostaje tam zapisany napis Java
2. kompilator zauważa, że znowu definiujemy Stringa o takiej samej zawartości i nie ma zamiaru rezerwować kolejnego miejsca w pamięci, w której będzie siedziała taka sama zawartość i dlatego zmienna napis2 będzie wskazywała na to samo co zmienna napis1

Dlaczego kompilator nie chce dawać nowego miejsca w pamięci? Ze względu na oszczędność. A co będzie jeśli zawartość Stringa się zmieni? A właśnie, że nie. Figa z makiem! String jest klasą niezmienniczą więc się nie zmieni. O tym napiszę później.

Taki system zapisu Stringów o tej samej zawartości nazywamy **String Common Pool**.

Przeanalizujmy jeszcze raz drugi przypadek

Tutaj mamy do czynienia z definiowaniem Stringa jak na klasę przystało, a więc przy użyciu słowa kluczowego new. Taki sposób definiowania Stringa sprawia, że jednej zmiennej i drugiej zmiennej czyli napis1 i napis2 będą przydzielone osobne miejsca w pamięci mimo tego, że będą w nich zapisane te same wartości.

Pozostaje jeszcze do rozwiązania jeden ważny problem. Dlaczego jeśli we właściwy sposób definiujemy String (czyli używając new) to jednak nie są one takie same mimo, że mają taką samą zawartość?

Musimy zapamiętać, że operator == porównuje typy proste, a do nich String nie należy. String jest klasą i jak na klasę przystało powinniśmy używać metody equals().

Zapamiętajmy : ze względu na to, że **String jest klasą**, to do porównywania obiektów klasy String **należy używać equals()**. Poniższy program działa już prawidłowo.

```
String napis1="Java";
String napis2="Java";
if (napis1.equals(napis2)) System.out.println("Takie same");
else System.out.println("Różne");
```

Jak zauważyliśmy pojawia się coś nowego (przed equals), a dokładniej operator . czyli operator **kropka**. Oznacza on dostęp do obiektu. Jest to **operator dostępu do obiektu**. Pisząc kropkę wyświetlaą się nam wszystkie czynności, które możemy na obiekcie klasy String wykonać. Wśród popularnych metod (czyli czynności) znajdziemy:

1. equals() – porównywanie dwóch stringów
2. lenght() – długość stringa
3. startsWith(ciag) - czy string zaczyna się określonym ciągiem znaków
4. endsWith(ciag) - czy string kończy się określonym ciągiem znaków

5.charAt(indeks) – zwraca element a więc znak, który znajduje się na pozycji indeks w stringu

Jak widać te metody zapisuje się w taki sposób, że na końcu znajdują się nawiasy okrągłe. Zawsze będzie to oznaczało, że mamy do czynienia z metodą czyli czynnością, którą można wykonać na obiekcie. Nawiasy nie zawsze będą puste. Jak widać powyżej czasami potrzebne są argumenty, aby wykonać jakąś metodę. Przykładowo charAt() nie może być pusty, bo nie będzie wiadomo, który znak zwrócić.

Jeszcze jedna ważna rzecz. Ostatnio napisałem, że String jest klasą niezmienniczą. Oznacza to, że nie można zmienić jego elementów czyli znaków. Jak zatem działał program z pierwszych rozdziałów. Dla przypomnienia

```
"The sum is " + 18
```

Co tu się działo? Pamiętacie? Wynikiem było

```
"The sum is 18"
```

```
// tutaj poprawka StringBuilder
```

[Deklaracja łańcucha](#)

[Porównywanie dwóch liczb](#)

[Porównywanie dwóch łańcuchów](#)

Programowanie funkcyjne

Zasady

1. Nasze klasy powinny być immutable czyli dajemy **final** (nie będzie więc dziedziczenia).
2. Pola składowe klasy są **inicjalizowane w konstruktorze**.
3. **Nie tworzymy seterów.**
4. Aby przeczytać wartości pól składowych tworzymy **getery**.
5. Metody są czyste. Nie modyfikują stanów innych obiektów.
6. Jeśli modyfikujemy jednak obiekt, to zwracamy już inny obiekt, jego kopię.

7. Unikamy nulli. Lepiej zwrócić obiekt klasy Optional<Klasa> niż null.
8. Metody są tak samo ważne jak klasy. Mogą być parametrami innych metod, a także mogą być przypisywane do zmiennych lokalnych.

Co to jest interfejs funkcyjny?

To taki interfejs, który posiada **tylko jedną metodę**

Przykłady:

1. Metoda run() z interfejsu Runnable
2. Metoda compareTo() z interfejsu Comparable

Interfejsy funkcyjne najlepiej oznaczać adnotacją **@FunctionalInterface**.

Java 1.8 to kilkanaście nowych interfejsów, właśnie funkcyjnych

Wyrażenie lambda – od Java 1.8

Są to anonimowe metody, bo nie mają nazwy.

Mogą zastąpić każdy interfejs funkcyjny

(String s)->{sout (c);};

Runnable r=()-> sout (a);

Jeśli metoda coś zwraca, to i tak tego nie piszemy z prawej strony, bo domyślnie jest return.

Nie trzeba nawet pisać jak jest typ zmiennej z lewej. Java powinna się domyślić ;)

Klasa Student i klasa Indeks

```
final public class Indeks {  
    private String indexNumber;  
  
    public Indeks(String indexNumber) {  
        this.indexNumber = indexNumber;  
    }  
  
    public String getIndexNumber() {  
        return indexNumber;  
    }  
}
```

```

final public class Student {
    private String name;
    private int age;
    private Indeks index;

    public Student(String name, int age, String indexNumber) {
        this.name = name;
        this.age = age;
        this.index = new Indeks(indexNumber);
    }

    public String getName() {
        return name;
    }

    public int getAge() {
        return age;
    }

    public Indeks getIndex() {
        return index;
    }

    @Override
    public String toString() {
        return "Student{" +
            "name='" + name + '\'' +
            ", age=" + age +
            ", index='" + index +
            '\'';
    }
}

public Student changeIndexNumber(String indexNumber) {
    return new Student("Paweł", 39, indexNumber);
}
}

```

Interfejs Predicate i metoda test(T)

1. Pomaga przy sekwencyjnym przetwarzaniu danych.
2. Na podstawie obiektu zwraca nam Booleana.

Klasa App

```

public class App {
    public static List<Student> createData() {
        List<Student> result=new ArrayList<>(
            Arrays.asList(
                new Student("Paweł", 38, "123"),
                new Student("Jacek", 34, "345"),
                new Student("Kasia", 26, "341"),
                new Student("Tomasz", 39, "3145")
            )
        );
    }
}

```

```
        return result;
    }
}
```

```
public static void main(String[] args) {
    List<Student> students=createData();
    Predicate<Student> kasiaPredicate=new Predicate<Student>() {
        @Override
        public boolean test(Student student) {
            return student.getName().equals("Kasia");
        }
    };
    System.out.println(students);
}
```

W metodzie main tworzymy listę studentów i predykat kasiaPredicate, który pomoże nam wyszukać studentów o imieniu Kasia. Ten zapis można zastąpić następującym wyrażeniem lambda

```
Predicate<Student> kasiaPredicate=student ->
student.getName().equals("Kasia");
```

A więc będzie

```
public static void main(String[] args) {
    List<Student> students=createData();
    Predicate<Student> kasiaPredicate=student ->
student.getName().equals("Kasia");
    System.out.println(students);
}
```

Napiszemy metodę, która dla listy studentów do niej przekazanej zwraca listę studentów spełniających test zawarty w predykacie.

```
public static List<Student>getStudents(List<Student>
list,Predicate<Student> predicate){
    List<Student> result=new ArrayList<>();
    for (Student student:list
         ) {
        if (predicate.test(student))result.add(student);
    }
    return result;
}
```

Obiekt spełnia test predykatu jeśli zostanie zaakceptowany przez metodę test(T)

Teraz wystarczy uruchomić tę metodę, aby otrzymać listę studentów, mających na imię Kasia.

```
System.out.println(getStudents(students,kasiaPredicate));
```

Jakie operacje można wykonywać na predykatach?

And, Or, negate

```
Predicate<Student> equals38andKasia=kasiaPredicate.and>equals38Predicate;
System.out.println(getStudents(students>equals38andKasia));
```

Wybierze nam studentów, którzy spełniają jednocześnie dwa warunki

Interfejs Consumer i metoda accept()

Na podstawie obiektu wykonuje jakąś operację dzięki metodzie accept() ale nic nie zwraca. Np. coś wyświetla.

Napiszemy teraz consumera, który na podstawie obiektu Student nic nie zwraca, ale wyświetla imię studenta.

```
Consumer<Student> studentNameConsumer=new Consumer<Student>() {  
    @Override  
    public void accept(Student student) {  
        System.out.println(student.getName());  
    }  
};
```

Można uprościć do wyrażenia lambda

```
Consumer<Student> print= student -> System.out.println(student.getName());
```

I wyświetlić (czyli skonsumować studentów)

```
public static void  
consumeStudents(List<Student>list,Consumer<Student>consumer){  
    for (Student student:list  
         ) {  
        consumer.accept(student);  
    }  
}
```

```
consumeStudents(students,print);
```

A jak zrobić skonsumowanie (wyświetlenie studentów po przefiltrowaniu?)

```
consumeStudents(getStudents(students,kasiaPredicate),print);
```

i wszystko jasne.

Jak się łączy consumerów? Metodą andThen(C)

```
Consumer<Student> printName= student ->  
System.out.println(student.getName());  
Consumer<Student> printAge=student -> System.out.println(student.getAge());  
Consumer<Student> two=printName.andThen(printAge);  
consumeStudents(getStudents(students,kasiaPredicate),two);
```

Należy pamiętać, że jeśli nie powiedzie się wykonywanie pierwszego consumera, to i drugi będzie zatrzymany.

Interfejs Supplier

Nie bierze żadnych argumentów, a zwraca obiekt czyli odwrotnie jak w przypadku Consumera.

Napiszemy teraz suppliera, aby zwracał nam całą listę studentów.

```
Supplier<List<Student>> supplyStudentList=() -> createData();
```

Albo jako zapis funkcyjny

```
Supplier<List<Student>> supplyStudentList= App::createData;
```

Teraz użyjemy suppliera razem z consumerem i predicate.

```
consumeStudents(getStudents(supplyStudentList, kasiaPredicate), two);
```

oczywiście aby to zadziałało należy trochę zmienić metodę getStudents()

```
public static List<Student> getStudents(Supplier<List<Student>>
supplier, Predicate<Student> predicate) {
    List<Student> result=new ArrayList<>();
    List<Student> students=supplier.get();
    for (Student student:students
         ) {
        if (predicate.test(student))result.add(student);
    }
    return result;
}
```

Interfejs Function i metoda apply()

Bierze jeden typ obiektu, modyfikuje i zwraca inny obiekt.

Zrobimy teraz Function, które będzie brało studenta i zwracało jego imię.

```
Function<Student, String> functionStudentName=student -> student.getName();
```

Albo prościej

```
Function<Student, String> functionStudentName= Student::getName;
```

Dodamy teraz funkcję do trzech pozostałych interfejsów.

Wcześniej musimy jednak zmienić consumera

```
Consumer<String> printString= string-> System.out.println(string);
```

```
consumeStudents(getStudents(supplyStudentList, kasiaPredicate), functionStudentName, printString);
```

```
public static void
consumeStudents(List<Student>list, Function<Student, String>
function, Consumer<String>consumer) {
for (Student student:list
     ) {
```

```
        consumer.accept(function.apply(student));  
    }  
}
```

i teraz będzie działać

Interfejs BiFunction

Bierze dwa obiekty, a zwraca trzy obiekty.

Interfejs BinaryOperator

Na podstawie dwóch obiektów tworzy jeden.

Warianty prymitywne interfejsów funkcyjnych

IntPredicate, na podstawie int zwraca Booleana

DoublePredicate, LongPredicate itd.

Podobnie jest z Consumerami i Supplierami.

Jest też np. KlasaToIntFunction = pobiera jakiś obiekt i zwraca inta. Są odpowiedniki dla innych typów prymitywnych.

Method references

To zapis np.:

```
Function<Student, String> functionStudentName= Student::getName;
```

Gdzie App jest naszą klasą.

Można więc powiedzieć, że jest to skrócenie zapisu lambdy.

Optional i metoda get()

Opakowuje inny obiekt i mówi o nim, że może być lub nie być.

Chodzi o to, by uniknąć zwracania nulli.

1 sposób, gdy nie jesteśmy pewni, czy nie będzie nullem

```
public Optional<Indeks>getIndex() {  
    return Optional.ofNullable(index);  
}
```

2 sposób, gdy jesteśmy pewni, że nie będzie nullem

```
public Optional<Indeks>getIndex() {  
    return Optional.of(index);  
}
```

3 sposób, gdy zwracamy pustą wartość

```
public Optional<Indeks> getIndex() {  
    return Optional.empty();  
}
```

Jak korzystać z Optional?

```
Optional<Indeks> indeks=supplyStudentList.get().get(0).getIndex();
```

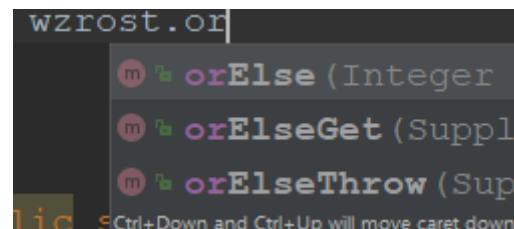
1 sposób

```
if(indeks.isPresent()) {  
    System.out.println(indeks.get().getIndexNumber());  
}
```

jeśli wywołamy geta na pustym Optional dostaniemy wyjątek

2 sposób (tutaj będzie consumer)

```
indeks.ifPresent(i ->System.out.println(i));
```



wzrost.or|
 m m **orElse** (Integer
 m m **orElseGet** (Suppl
 m m **orElseThrow** (Sup
lic SCtrl+Down and Ctrl+Up will move caret down

3 sposób (filtrowanie)

```
indeks.filter(i->i.getIndexNumber().equals("123")).ifPresent(i->  
    System.out.println(i.getIndexNumber()));
```

4 sposób (mapowanie)

```
indeks.map(i->i.getIndexNumber()).filter(number->  
    number.equals("123")).ifPresent(index-> System.out.println(index));
```

Stream API

To zestaw metod do strumieniowego przetwarzania danych.

Po streamie możemy przejechać się tylko raz czyli inaczej niż w kolekcjach.

Przykład

```
supplyStudentList.get().stream().filter(kasiaPredicate).map(Student::getName).forEach(printString);
```

Generowanie wartości dla strumieni

1 sposób (predefiniowane wartości)

```
Stream.of("A", "B", "C").forEach(printString);
```

2 sposób (kolekcje)

```
List<Student> studentList1=createData();  
studentList1.stream().filter(kasiaPredicate).forEach(printName);
```

3 sposób (kolekcje)

```
Stream.generate(supplyStudentList).forEach(print);
```

Tylko, że trzeba napisać jeszcze consumera dla listy studentów

4 sposób (Supplier)

```
Stream.generate(()->Math.random()).limit(10).forEach(System.out::println);
```

5 sposób (iterate, wyświetlenie 20 liczb parzystych, począwszy od 0)

```
Stream.iterate(0, i->i+2).limit(20).forEach(System.out::println);
```

6 sposób (liczby podzielne przez 2)

```
IntStream.rangeClosed(5, 100).filter(i->i%2==0).forEach(System.out::println);
```

Filtr

Filter to operacja pośrednia na strumieniach. Filter przyjmuje jako argument Predicate.

```
public static Stream<Student> createDataStream() {  
    Student student1=newStudent("Paweł", 38, "123");  
    Student student2=newStudent("Jacek", 34, "345");  
    Student student3=newStudent("Kasia", 38, "341");  
    Student student4=newStudent("Tomasz", 39, "3145");  
    return Stream.of(student1, student2, student3, student4);  
}
```

oraz metoda main()

```
public static void main(String[] args) {  
    Predicate<Student> over30Predicate = student -> student.getAge() > 30;  
    Consumer<String> println = System.out::println;  
    Function<Student, String> getStudentName = Student::getName;  
  
    createDataStream().filter(over30Predicate).map(getStudentName).forEach(prin  
    tln);  
}
```

Wyświetlamy więc imiona studentów powyżej 30 lat.

Map

Map to operacja pośrednia na strumieniach. Map przyjmuje jako argument Function.

Map i Filter można stosować wiele razy i na przemian.

ForEach

ForEach to metoda terminalna, która kończy używanie strumienia.

FindFirst, AnyMatch,AllMatch, NoneMatch

Wszystkie trzy to metody terminalne, które po wykonaniu swoich zadań zamykają strumień.

Wyświetlimy imię pierwszego studenta, którego wiek przekracza 30 lat

```
createDataStream().filter(over30Predicate).map(getStudentName).findFirst().  
ifPresent(System.out::println);
```

AnyMatch – czy dowolny obiekt w strumieniu spełnia Predicate

Czy jest jakiś student o imieniu Paweł?

```
System.out.println(createDataStream().map(getStudentName).anyMatch(s -  
> s.equals("Paweł")));
```

AllMatch – czy wszystkie obiekty spełniają określony warunek

Czy wszystkie imiona są palindromami

```
System.out.println(createDataStream().map(getStudentName).allMatch(s -> new  
StringBuilder(s).reverse().equals(s)));
```

Czy wszystkie wyrazy znajdujące się w pliku są palindromami?

```
Files.readAllLines(Paths.get("palindrom.txt")).stream().map(String::toLowerCase  
Case).allMatch(s -> new StringBuilder(s).reverse().equals(s));
```

Reduce

Metoda terminalna. Redukuje strumień do jednej wartości.

Można użyć do wyszukiwania min,max czy do łączenia stringów.

Sumowanie dziesięciu liczb losowych (użycie jako drugiego parametru newBinaryOperator i przekształcić na lambdę)

```
System.out.println(Stream.generate(Math::random).limit(10).reduce(0.0, (aDouble, aDouble2) -> aDouble+aDouble2));
```

Jak znaleźć najstarszego studenta?

```
createDataStream().map(s->s.getAge()).max(Comparator.naturalOrder()).ifPresent(System.out::println);
```

lub

```
createDataStream().map(Student::getAge).reduce(Integer::max).ifPresent(System.out::println);
```

Collect

Metoda terminalna, jest to specjalny typ reduce, który pozwala nam np. na zebranie wszystkich elementów w listę.

Jak uzyskać listę wieku studentów?

```
System.out.println(createDataStream().map(Student::getAge).collect(Collectors.toList()));
```

Counting – zwraca liczbę elementów.

Jak połączyć wszystkie elementy ze strumienia do jednego stringa z separatorem „ , ” ?

```
createDataStream().map(Student::getAge).map(s->s.toString()).collect(Collectors.joining(", "))
```

Jak utworzyć mapę, klucz = age, wartość = liczba studentów w danym wieku

```
Map<Integer, List<Student>> list=createDataStream().collect(Collectors.groupingBy(Student::getAge)); System.out.println(list);
```

Limit, skip, distinct, sorted, count

Limit – ograniczenie do liczby elementów

Skip – pomija określoną liczbę elementów

Distinct – bierzemy pod uwagę tylko różne obiekty (hashcode i equals)

Sorted() – sortowanie wg naturalnego porządku lub newComparator w nawiasie

Count – oblicza ilość elementów w strumieniu (reduktor)

Strumienie typów prymitywnych

Zaleta?

Szybciej pracuje się na strumieniach prymitywnych

```
IntStreamintStream=createDataStream().map(Student::getAge).mapToInt(value ->value.intValue());
```

Mожет быть еще только Long и Double

Wada?

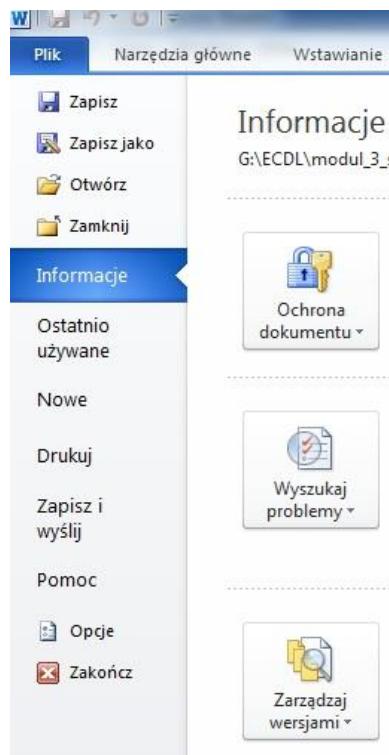
Od teraz wszędzie w parametrach trzeba podawać Integerowe odmiany interfejsów funkcyjnych.

Edytor tekstu – MS Word

Podstawowe informacje

Aby uruchomić edytor tekstu należy wybrać odpowiednią ikonę z menu start o nazwie Microsoft Word. Do uruchomienia edytora tekstu Microsoft Word wystarczy również otwarcie pliku wcześniej w nim zapisanego.

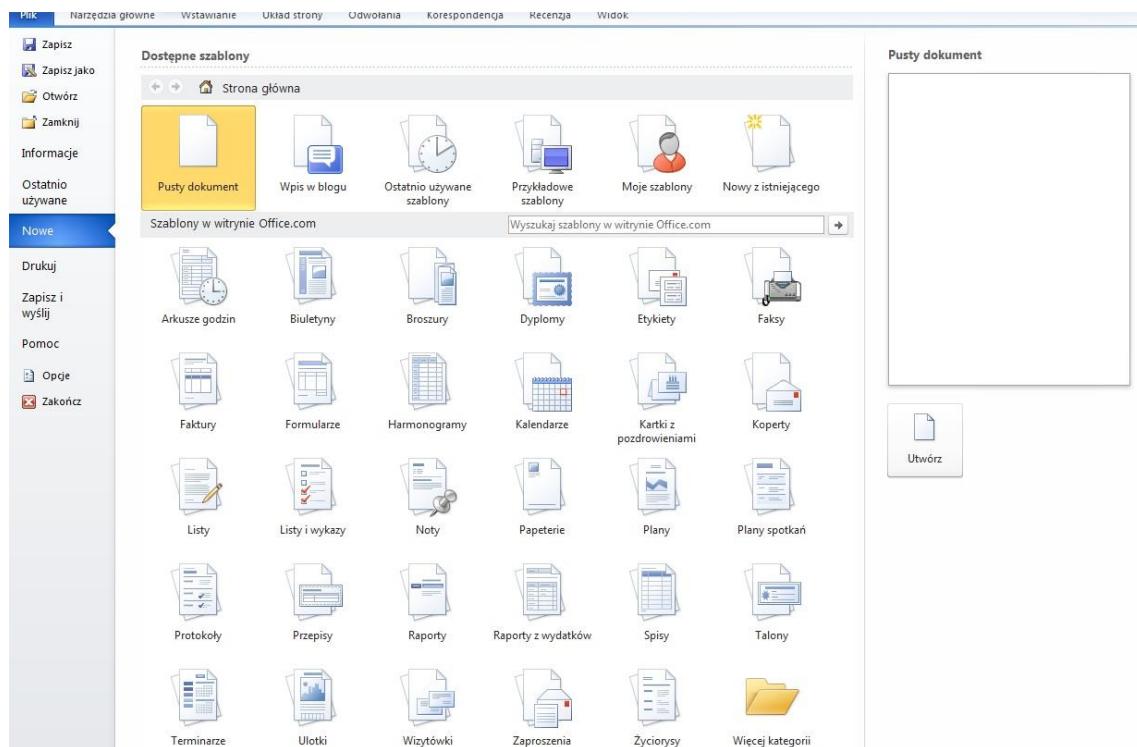
Aby zakończyć pracę z edytorem tekstu Microsoft Word należy z menu górnego Plik wybrać opcję zakończ.



Aby **otworzyć dokument** w edytorze należy wybrać opcję **Otwórz** z widocznego na powyższym obrazku menu Plik.

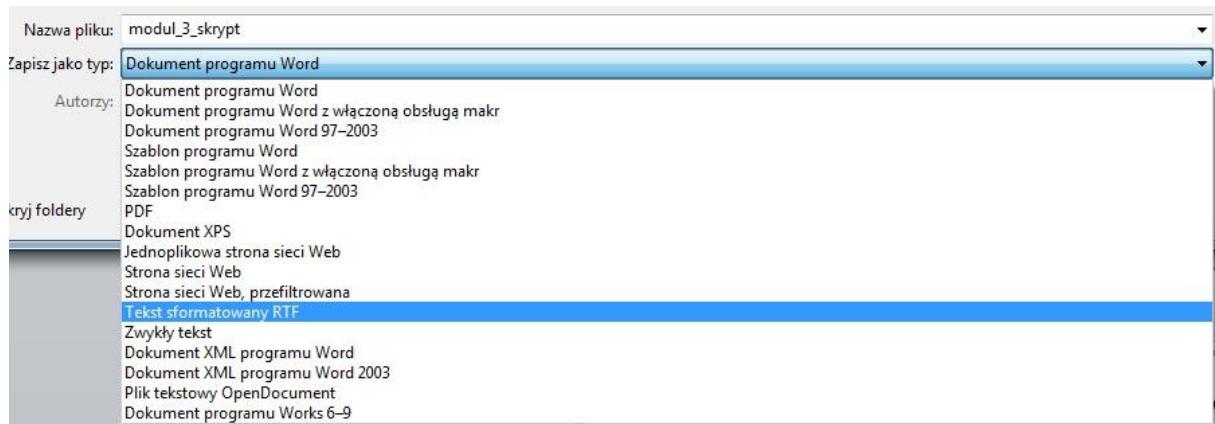
Aby **zakończyć pracę z dokumentem** należy wybrać opcję **Zamknij**.

Aby utworzyć dokument w oparciu o **domyślny szablon** należy z menu plik wybrać opcję **nowe**. Na poniższym obrazku widać dostępne szablony.



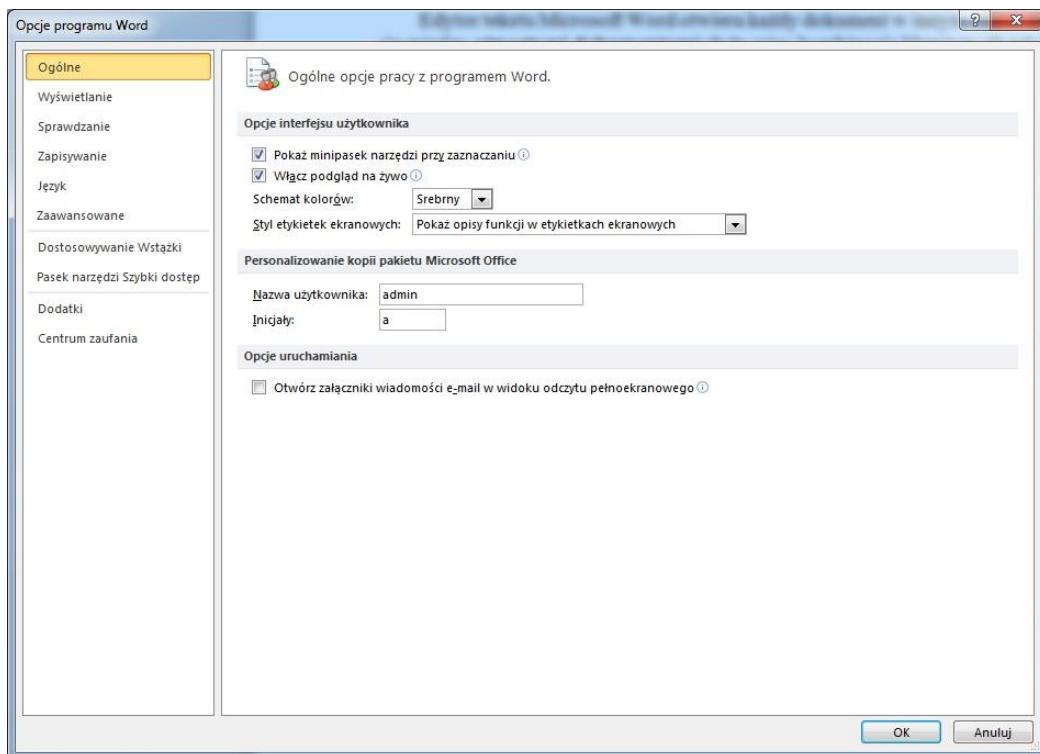
Aby **zapisać plik** na dysku z domyślną lub pod określoną nazwą należy użyć opcji zapisz jako z menu Plik.

Po wybraniu opcji zapisz jako z menu plik i po rozwinięciu **zapisz jako typ** pojawia się nam możliwość zapisania naszego dokumentu jako pliku innego typu.



Edytor tekstu Microsoft Word otwiera każdy dokument w innym oknie. Do poruszania się między **otwartymi dokumentami** służy więc kombinacja klawiszy ALT-TAB lub przyciski maksymalizacji i minimalizacji okna.

Do ustawianie podstawowych preferencji dla edytora tekstu służy aplet **opcje programu Word**. Aby go uruchomić należy z menu plik wybrać opcję „opcje”. Wtedy zobaczymy

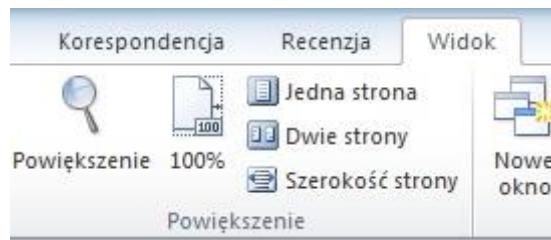


Zauważmy od razu, że możemy tutaj wybrać **domyślną nazwę użytkownika** dla tworzonych dokumentów („Ogólne->Personalizowanie kopii pakietu Microsoft Office->nazwa użytkownika”).

Podobnie **domyślny folder zapisu naszych dokumentów** znajdziemy w „Zapisywanie->Domyślna lokalizacja pliku”.

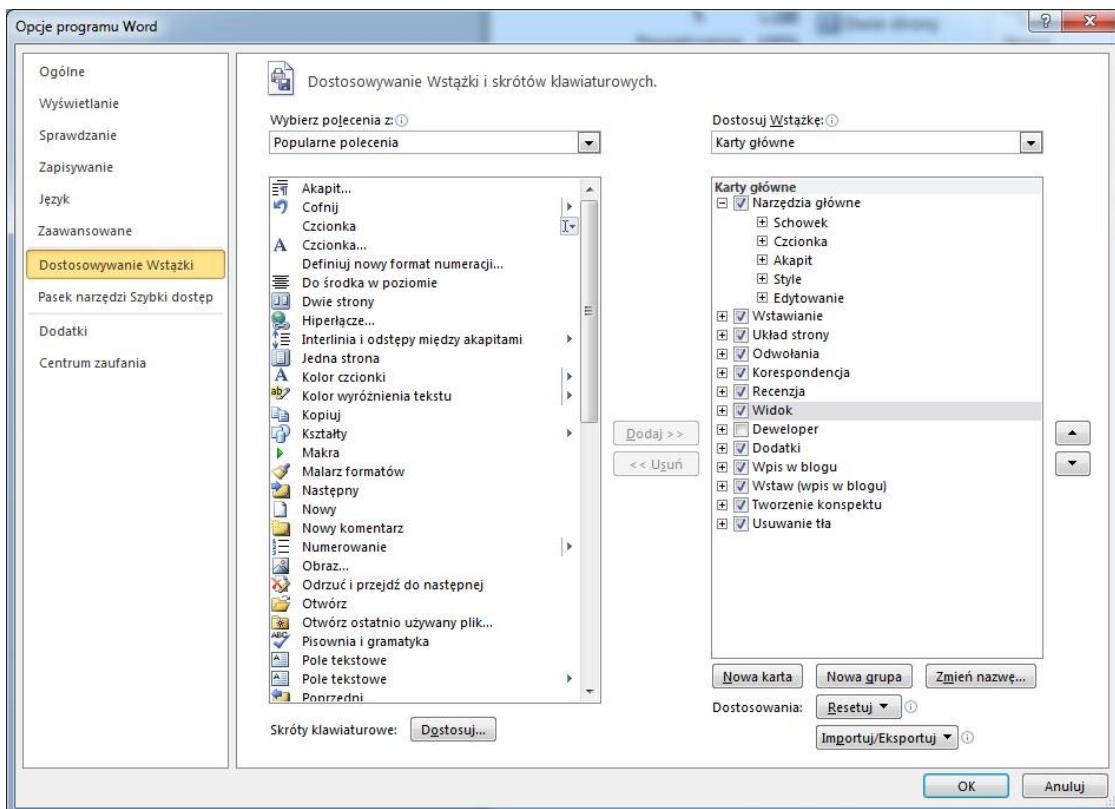
Informacje na temat pomocy znajdziemy wybierając opcję **pomoc** z menu Plik.

Aby **powiększyć wyświetlanie dokumentu** należy wybrać z menu górnego widok. Wtedy zobaczymy



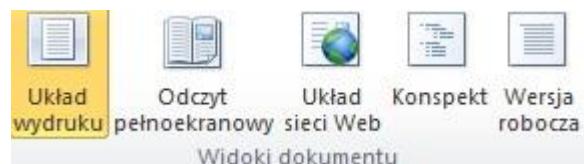
Drugi sposób – szybszy – powiększania widoku to naciśnięcie klawisza CTRL i przesuwanie scrolla myszki.

Aby dostosować widoczność i dostępność pasków narzędziowych i wstążki należy ustawić odpowiednie opcje w **Opcje programu Word->dostosowywanie wstążki**

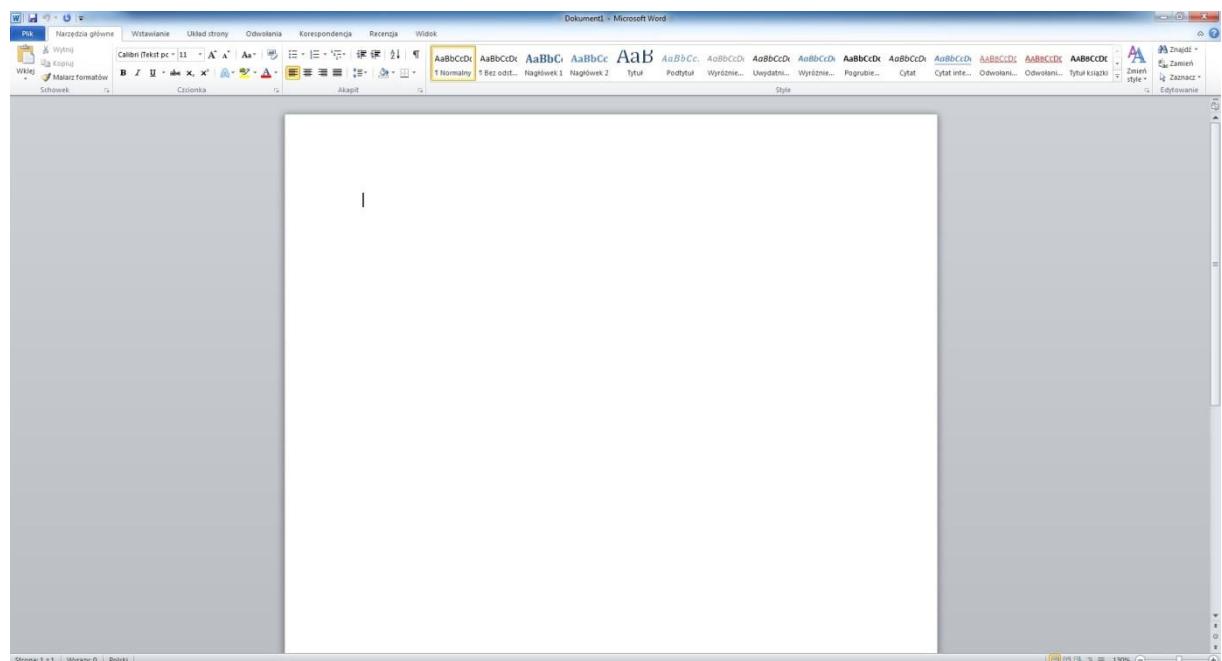


Tworzenie dokumentu

Aby zmienić tryb wyświetlania dokumentu należy wybrać odpowiednią opcję z menu widok->widoki dokumentu tak jak na poniższym obrazku

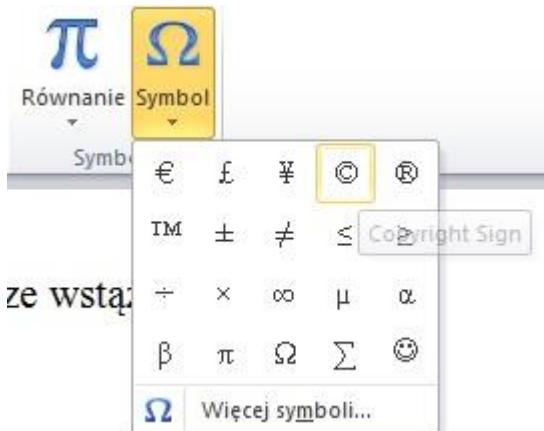


Tekst do dokumentu wprowadzamy w białym pustym polu.



O gotowości edytora do wprowadzania tekstu świadczy obecność czarnego migotającego kurSORA.

Aby wprowadzić symbole i znaki specjalne należy wybrać ze wstążki **wstawianie->symbole**. Po kliknięciu na „symbole” zobaczymy symbole i znaki specjalne, które możemy wstawić.



Aby zobaczyć wszelkie znaki formatowania należy użyć klawisza pokaż wszystko

¶

Znajdziemy go na wstążce, pasek **narzędzia główne->akapit.**

Wtedy zamiast przykładowego tekstu poniżej

Tutaj jest tabulator

Tutaj spacja

Tutaj kilka spacji|

zobaczymy ukryte formatowanie

Tutaj → jest → tabulator ¶

Tutaj.spacja ¶

Tutaj....kilka....spacji ¶

W celu **przesunięcia się o jeden znak** w prawo w edytowanych tekście należy nacisnąć **kursor w prawo, w lewo** to kursor w lewo. Jeśli chcemy **poruszać się o całe słowa** w lewo lub prawo należy oprócz kursora kierunku trzymać wciśnięty klawisz **CTRL**. Jeśli chcemy poruszać się **po akapitach**, to należy używać **kursorów w górę i w dół**, trzymając wciśnięty klawisz **CTRL**.

Jeśli do powyższych kombinacji klawiszy dodamy klawisz **SHIFT**, który będziemy trzymać, to wtedy wraz z poruszaniem się po dokumencie będziemy **zaznaczać tekst**. Jeśli chcemy zaznaczyć **cały dokument** należy nacisnąć **CTRL-A**.

Edytor tekstu Microsoft Word oferuje dwa tryby wprowadzania tekstu: **wprowadzanie i zastępowanie**.

Domyślnym trybem pracy jest wprowadzanie. W tym trybie znaki wprowadzane z klawiatury powodują „rozsuwanie” się istniejącego już wprowadzonego tekstu. Drugim trybem jest zastępowanie, w którym każda wprowadzona przez nas literka zastępuje aktualną.

Aby przełączyć się pomiędzy tymi trybami należy użyć opcji programu Word i wybrać „**Zaawansowane>Użyj klawisza Insert do sterowania trybem zastępowania**”.

Do edycji tekstu bardzo ważne będą następujące klawisze:

- a) backspace - usuwa jeden znak na lewo od kurSORA
- b) ctrl-backspace - usuwa jeden wyraz na lewo od kurSORA
- c) delete - usuwa jeden znak na prawo od kurSORA
- d) ctrl-delete - usuwa jeden wyraz na prawo od kurSORA

Aby wprowadzić wielką literę należy użyć klawisza **SHIFT**. Jeśli będziemy pisać dużą „porcję” tekstu wielkimi literami należy rozważyć włącznie klawisza **CAPS-LOCK**.

Aby wyszukać określone słowo w dokumencie należy wybrać **Narzędzia główne->Edytowanie->Znajdź** lub nacisnąć kombinację klawiszy **CTRL-F**.

Aby zastąpić jedno słowo drugim należy wybrać **Narzędzia główne->Edytowanie->Zamień** lub nacisnąć kombinację klawiszy **CTRL-H**.

Aby skopiować tekst należy najpierw go zaznaczyć. Następnie należy wcisnąć prawy klawisz myszki i z menu kontekstowego wybrać **Kopiuj**. Teraz należy wybrać miejsce docelowe, ustawiając odpowiednio kurSOR, nacisnąć prawy klawisz myszki i wybrać **Wklej**.

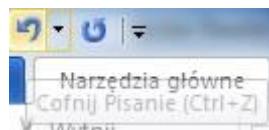
Aby przenieść tekst należy postępować podobnie jak powyżej w przypadku kopiowania z tą różnicą, że zamiast **Kopiuj** należy wybrać **Wytnij**.

UWAGA:

Zamiast **Kopiuj** można nacisnąć **CTRL-C**, **Wytnij CTRL-X**, **Wklej CTRL-V**.

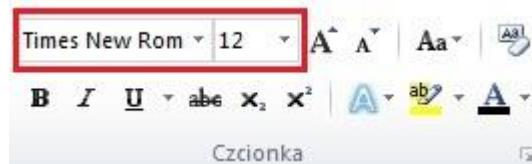
Aby usunąć tekst należy go najpierw zaznaczyć, a następnie nacisnąć klawisz **Delete** lub z menu kontekstowego wybrać **Wytnij**.

Aby **Cofnąć** poprzednią operację albo wykonać **Ponów** należy skorzystać z paska szybkiego dostępu.



Formatowanie

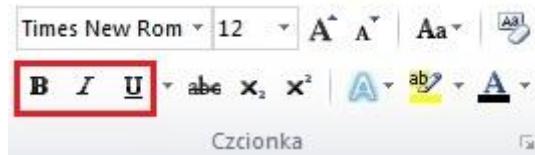
Zmiana formatu tekstu(kroju i wielkości czcionki) jest dostępna na wstążce na pasku Narzędzia główne ->Czcionka.



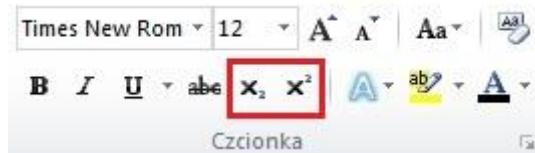
Wielkość czcionki można również zmienić w menu kontekstowym, które otwieramy prawym klawiszem myszki po zaznaczeniu tekstu.

Zmianę stylu czcionki dokonujemy w tym samym miejscu co zmianę formatu tekstu.

Aby napisać tekst czcionką: **pogrubioną** wybierzemy klawisz **B**, **pochyloną** **I**, a **podkreślona** klawisz **U**.

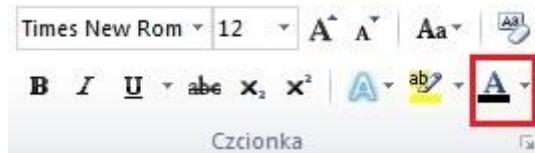


Aby dany wyraz bądź liczbę umieścić w **indeksie górnym lub dolnym** należy skorzystać z tego samego „menu”. Indeksy znajdują się tuż obok zmiany stylu czcionki.

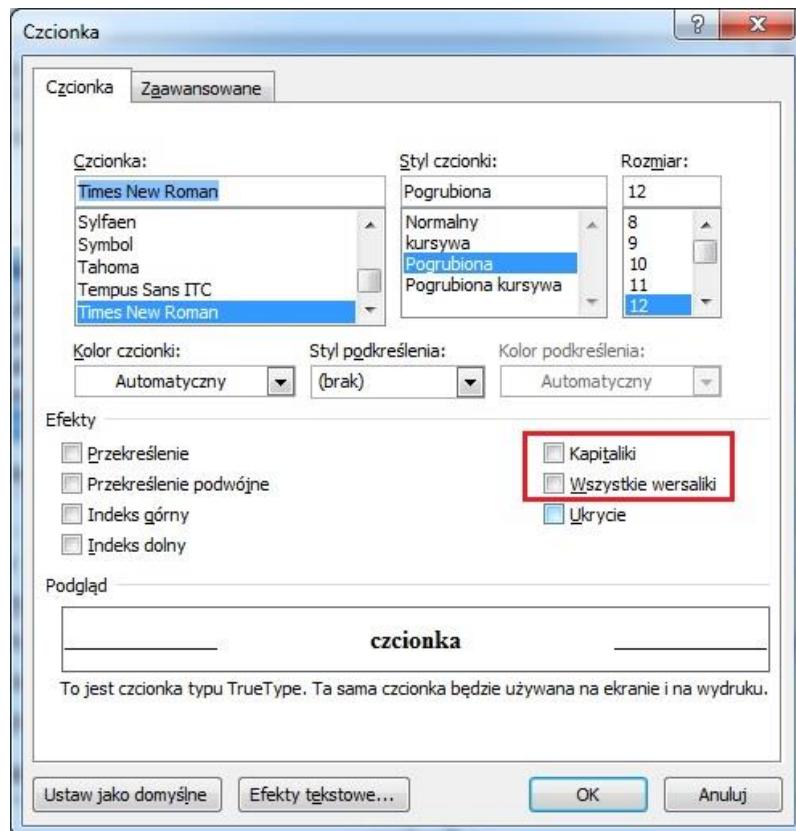


Na powyższym obrazku indeksy zaznaczono w czerwonej obwódkie.

Kolor czcionki można zmienić w miejscu zaznaczonym na czerwono na poniższym obrazku.



Kapitaliki i wersaliki dla pisanego tekstu można włączyć wybierając z menu kontekstowego opcję **czcionka**



Aby zastosować **automatyczne dzielenie wyrazów** należy z paska **Układ strony** wybrać **Dzielenie wyrazów->Automatycznie**.

Akapity

Akapit jest podstawowym blokiem tekstu rozpoczynającym się od nowego wiersza i kończącym się znakiem końca akapitu. Koniec akapitu uzyskuje się przez naciśnięcie klawisz ENTER. Podczas formatowania tekstu akapit traktowany jest jako odrębna całość. Akapity można podejrzeć przez użycie klawisza opisanego w 2.2.1 o nazwie **pokaż wszystko** (Narzędzia główne -> Akapit). Następny akapit zaczyna się tam gdzie kończy się poprzedni. Znakiem oddzielającym akapitu jest więc ¶, który widać po włączeniu **pokaż wszystko**.

Wiemy już, że wciśnięty klawisz ENTER powoduje utworzenie nowego akapitu, jednocześnie kończąc poprzedni. Jak w takim razie wymusić przejście do nowej linii bez tworzenia nowego akapitu? Należy zastosować „**miękkiego enter**”. Otrzymujemy go przez naciśnięcie kombinacji klawiszy

SHIFT-ENTER. W ten sposób przejdziemy do nowej linii bez tworzenia akapitu. Znak nowej linii wewnątrz akapitu można zobaczyć poniżej.

To·jest·akapit·numer·jeden.¶

To·jest·akapit·numer·dwa·a·tutaj·←
przejście·do·nowej·linii·ale·ciągle·ten·sam·akapit¶

Aby usunąć znak przejścia do nowej linii należy wcisnąć klawisz **pokaż wszystko**, a następnie pożądany znak.

Do dobrych praktyk w wyrównywaniu tekstu należą: **używanie wyrównywania, stosowanie wcięć** oraz **stosowanie tabulatorów zamiast spacji**.

Domylnym sposobem formatowania akapitu jest formatowanie do lewej. Aby zmienić formatowanie akapitu należy użyć **Narzędzia główne->Akapit**.



Te cztery elementy otoczone czerwoną obwódką pozwalają na **wyrównywanie tekstu: do lewej, wyśrodkowanie, do prawej i wyjustowanie** odpowiednio.

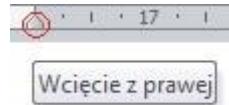
Wcięcie określa odległość akapitu od lewego lub prawego marginesu. Najlepszym sposobem ustawiania wcięć akapitów jest użycie znaczników znajdujących się na linijce poziomej.

Najpierw trzeba jednak włączyć pokazywanie linijkę. W tym celu wybierzemy **Widok->Pokazywanie->Linijka**.

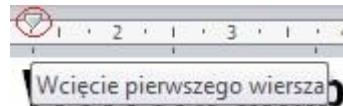
Aby wyrównać akapit z lewej strony należy znaczyć akapit, a następnie wybrać „**wcięcie z lewej**”, przesunąć go w pożądane miejsce.



Aby wyrównać akapit z prawej strony należy zaznaczyć akapit, a następnie wybrać „**wcięcie z prawej**”, przesunąć go w pożądane miejsce.



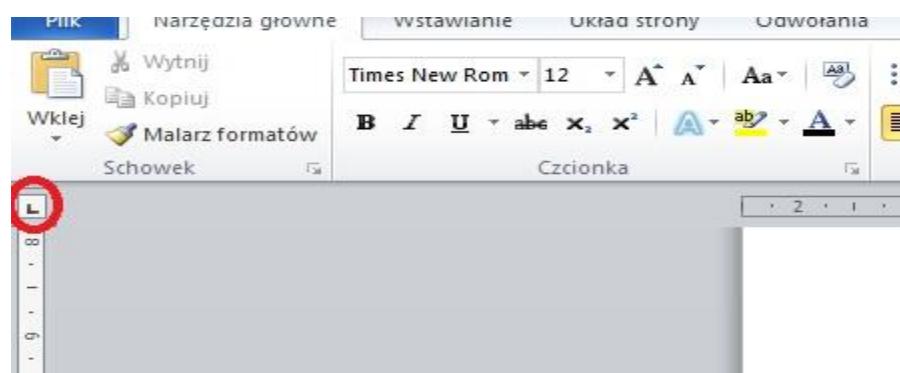
Aby zastosować wcięcie dla pierwszego wiersza należy zaznaczyć akapit, a następnie wybrać „**wcięcie pierwszego wiersza**”, przesunąć go w pożądane miejsce.



Najważniejszymi tabulatorami używanymi w tekście są :

- a) **tabulator lewy** – wpisywany tekst przesuwa się w prawo
- b) **tabulator środkowy** – wpisywany tekst jest wyśrodkowywany względem miejsca położenia tabulatora
- c) **tabulator prawy** – wpisywany tekst przesuwa się w lewo
- d) **tabulator dziesiętny** - miejsce dziesiętne liczb wpisywanych w kolejnych wierszach wypada zawsze w miejscu położenia tabulatora, niezależnie od tego ile miejsc dziesiętnych mają poszczególne liczby

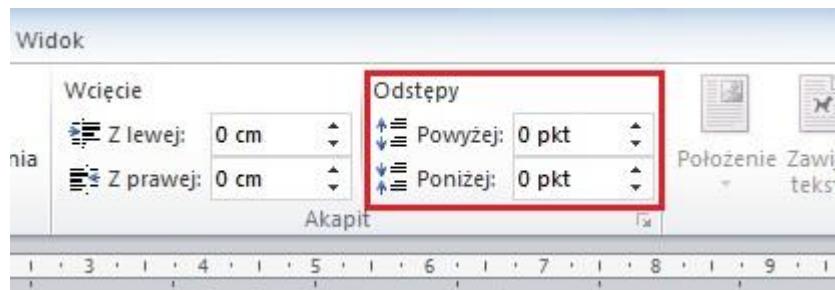
Aby użyć tabulatora należy najpierw wybrać jego odpowiedni rodzaj klikając na ikonę.



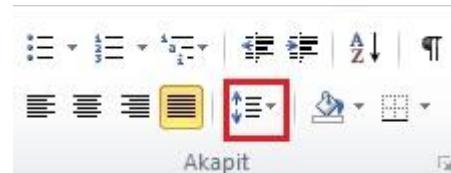
Następnie przenosimy go na linijkę poziomą, klikając w odpowiednim miejscu. Później wybieramy wiersz, w którym chcemy zastosować tabulator i wciskamy klawisz TAB. Teraz możemy pisać tekst.

Do oddzielania akapitów najlepiej posłużyć się odstępami zamiast stosować klawisze ENTER. Stosuje się tutaj odstępy przed i po akapicie.

Aby ustawić **odstępy między akapitami**, a więc przed akapitem i/lub po akapicie należy ustawić odpowiednią wielkość w **Układ strony->Akapit->Odstępy**.



Można również ustawić **odstępy między wierszami** wewnętrz akapitów. Aby to zrobić należy wybrać **Narzędzia główne->Akapit->Interlinia** i ustawić określona wartość.

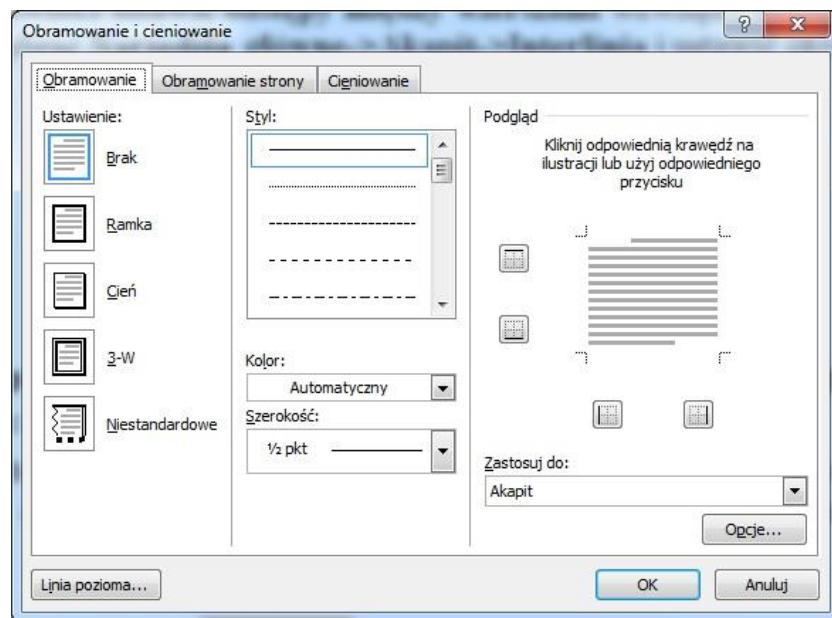


Aby zastosować **punktowanie** lub **numerowanie** dla przygotowanej listy należy zaznaczyć listę a następnie wybrać punktowanie lub numerowanie z **Narzędzia główne->Akapit**.



Zauważmy od razu, że obok tych przycisków mamy listę rozwijaną. Tam możemy ustawić sobie **niestandardowy sposób** przedstawiania listy punktowanej lub numerowanej.

Aby zastosować **obramowanie akapitu** należy zaznaczyć akapit, następnie wybrać **Układ strony->Tło strony->Obramowania stron**. Pokaż się okienko.



Należy wybrać zakładkę obramowanie. Tutaj można ustalić **obramowanie dla akapitu**. Należy jednak pamiętać aby w polu **Zastosuj do** wybrać **Akapit**.

Ten akapit jest obramowany.

Aby wybrać **cieniowanie dla akapitu** należy użyć zakładki cieniowanie, którą widać na poprzednim obrazku.

W tym akapicie zastosowano cieniowanie wraz z kolorem.

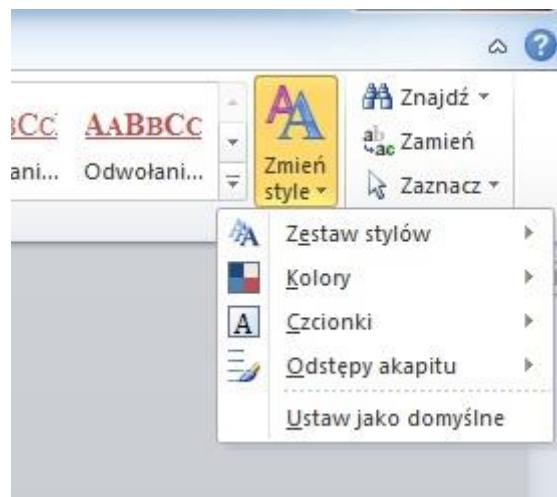
Style

Style są sposobem na uniknięcie konieczności **wielokrotnego formatowania w identyczny sposób** różnych elementów tekstu, np. akapitów. Style określają w ścisły sposób wygląd różnych elementów dokumentu takich jak tekst podstawowy czy nagłówki.

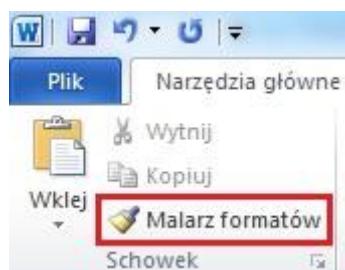
Aby zastosować istniejący **styl czcionki** do naszego tekstu należy najpierw zaznaczyć blok tekstu a później z paska **Style** wybrać odpowiedni styl.



Style akapitu stosuje się podobnie jak style czcionki. Dostępne style można podejrzeć po kliknięciu **Zmień style** tak jak na obrazku poniżej

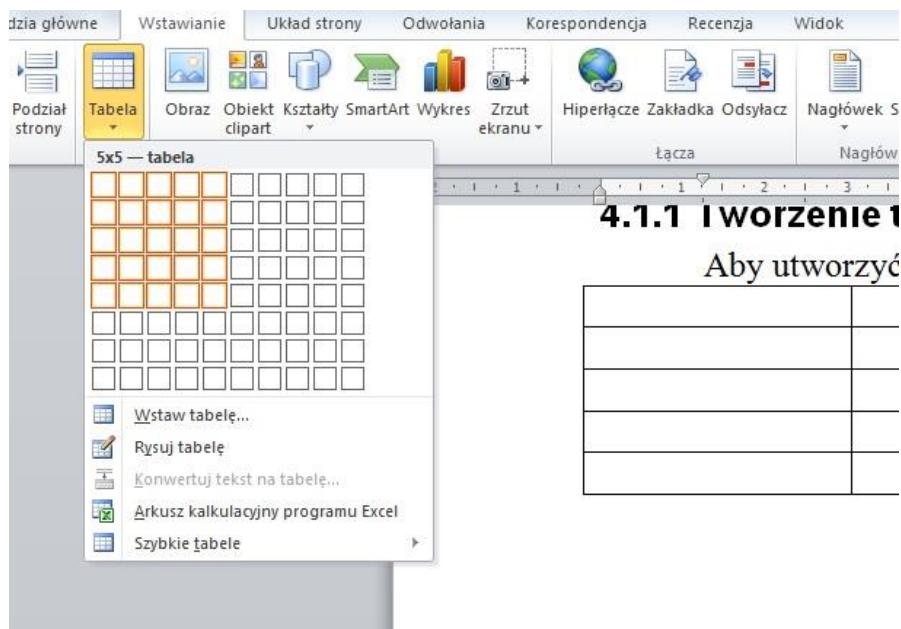


Wyobraźmy sobie, że piszemy tekst i stosujemy w nim skomplikowane formatowanie. Chcemy to formatowanie przenieść na inne fragmenty tekstu. Wtedy wykorzystamy **Malarza Formatów**. Aby to uczynić należy najpierw zaznaczyć tekst, z którego ma zostać pobrane formatowanie. Następnie kliknąć na przycisku Malarz formatów. Dalej trzeba zaznaczyć tekst, do którego ma zostać „wklejone” formatowanie.

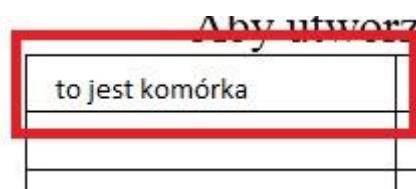


Obiekty

Aby **utworzyć tabelę** należy wybrać ze wstążki **Wstawianie->Tabela** i postępować zgodnie z instrukcjami.

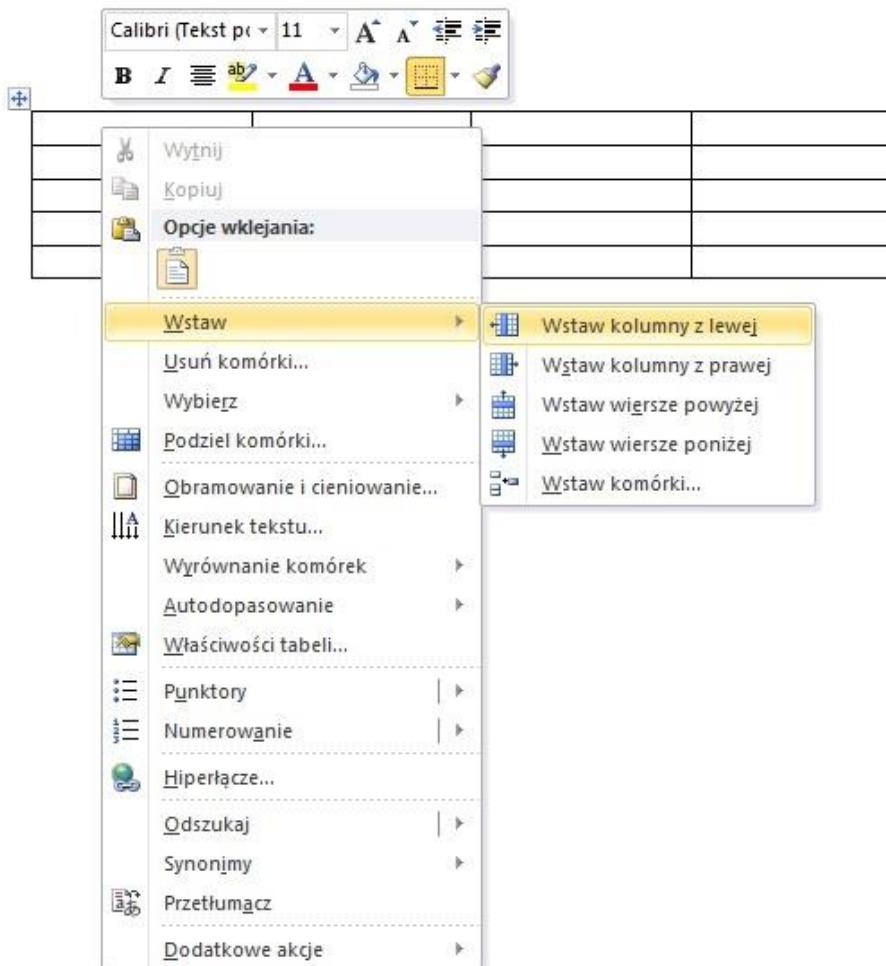


Kiedy mamy już tabelę, można zacząć **wpisywać do niej dane**. Dane wpisujemy do komórek tak jak zwykły tekst.

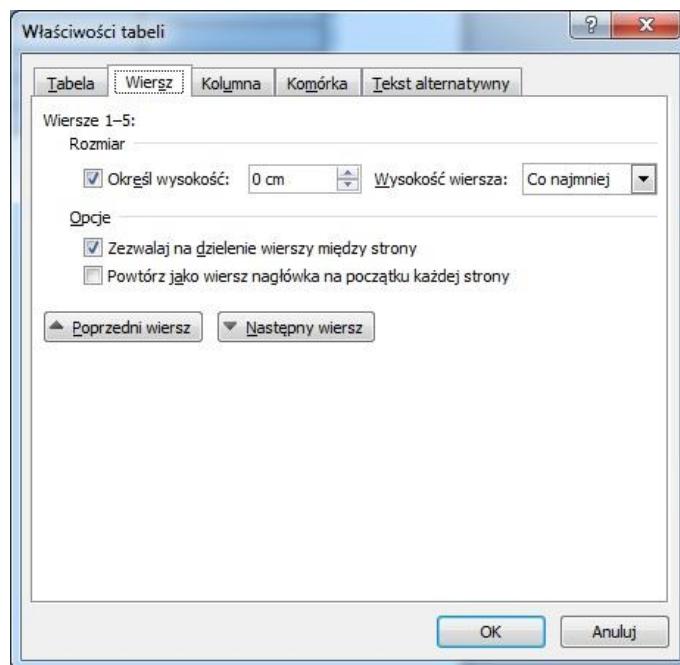


W celu **zaznaczenia elementów tabeli** należy użyć klawisza myszy i przeciągnąć na obszar, który ma być zaznaczony. Jeżeli chcemy zaznaczyć oprócz wierszy również kolumny, to należy użyć klawisza **CTRL**.

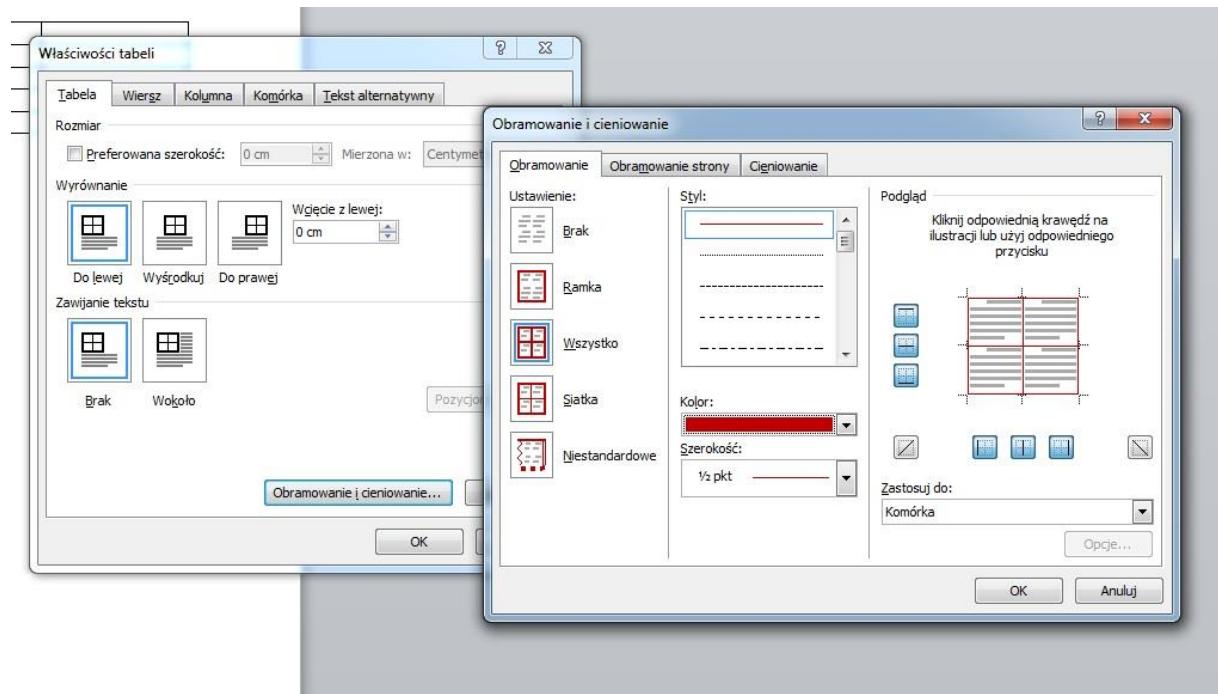
Aby **wstawić lub usunąć wiersz lub kolumnę** należy ustawić cursor w pożądanym miejscu tabeli (odpowiedniej komórce) i z menu kontekstowego wybrać **Wstaw** lub **Usuń komórki**.



Aby zmienić **szerokość kolumny i wysokość wiersza** najlepiej będzie zaznaczyć tabelę, wybrać z menu kontekstowego Właściwości tabeli. Odpowiedź na zadanie znajdziemy w zakładkach **Wiersz i Kolumna**.



Na poniższym obrazku we właściwościach tabeli widzimy zakładkę **Tabela**, a pod spodem klawisz z napisem **Obramowanie i cieniowanie**. Jeśli go klikniemy, to przejdziemy do kolejnego ekranu, który pozwoli nam zmienić ustawienia obramowania dla komórki tabeli.



Na poprzednim obrazku w okienku **Obramowanie i cieniowanie** widzimy zakładkę **Cieniowanie**. To tam można zmienić ustawienia cieniowania/koloru tła dla komórek tabeli.

Obiekty graficzne

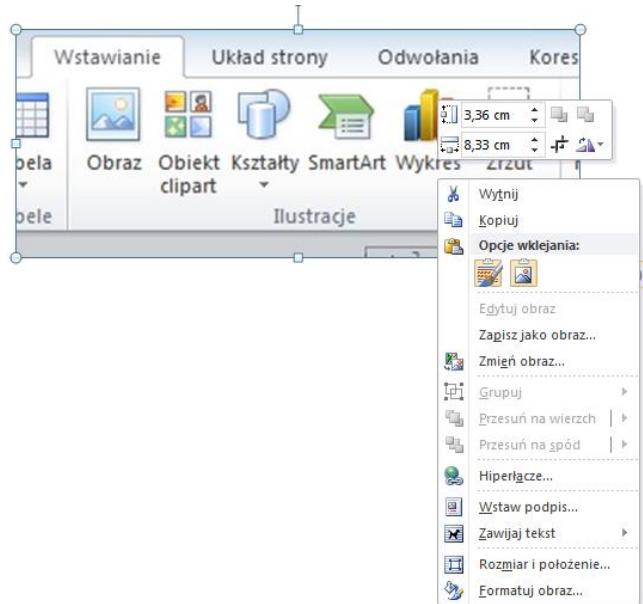
Obiektem może być **obraz, rysunek, grafika, wykres**. Aby wstawić taki obiekt do dokumentu należy wybrać ze wstążki **Wstawianie**, a następnie pożądany obiekt.



Obiekt **zaznaczamy klikając na niego myszką**.

Aby **skopiować** obiekt należy go **zaznaczyć myszką** i z menu kontekstowego wybrać **kopiuj**.
Aby **przenieść** obiekt należy go **zaznaczyć myszką i nie puszczaając lewego klawisza przenieść** w odpowiednie miejsce.

Aby **zmienić rozmiar obiektu** należy do zaznaczyć a następnie z menu kontekstowego wybrać **Formatuj obraz** i postępować według wskazówek.



Drugi sposób to chwycenie za róg obiektu i zmniejszenie go do pożądanej wielkości.

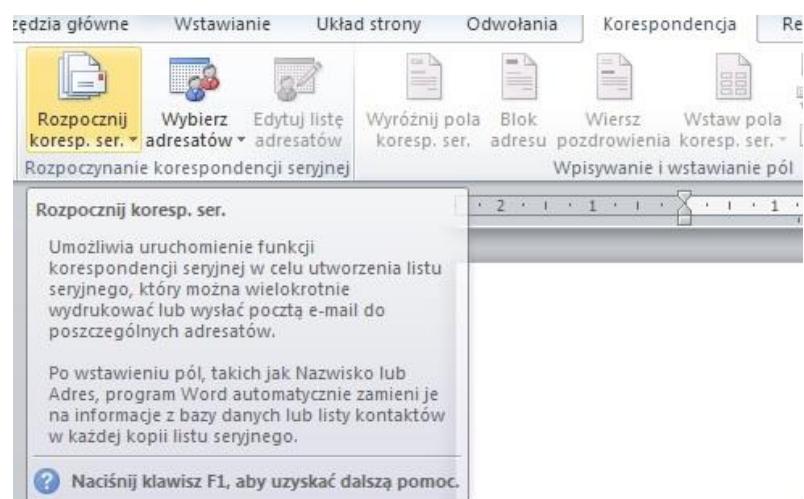
Aby usunąć obiekt, należy go zaznaczyć i nacisnąć klawisz **DELETE**.

Korespondencja seryjna

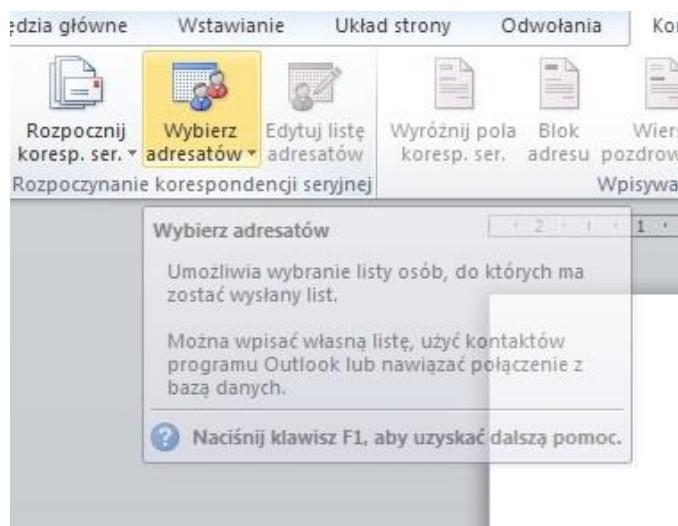
Korespondencja seryjna przydaje się nam kiedy chcemy **nапisać identyczne listy** (e-maile, dokumenty i inne) **do większej liczby osób**, a nie chcemy tworzyć dokumentu dla każdej osoby oddzielnie.

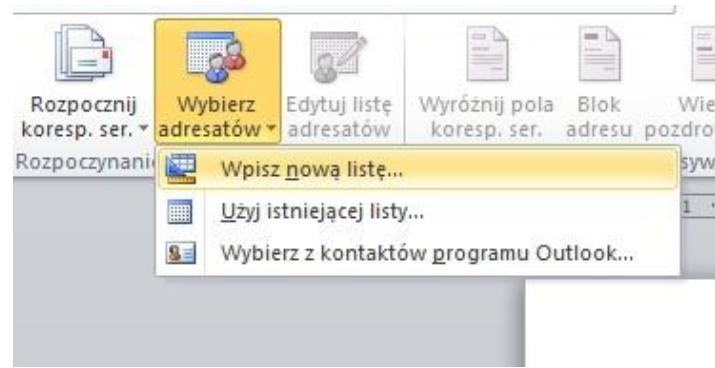
Pierwszym krokiem jest przygotowanie dokumentu głównego.

W tym celu wybieramy z paska Korespondencja klawisz Rozpocznij korespondencję seryjną.

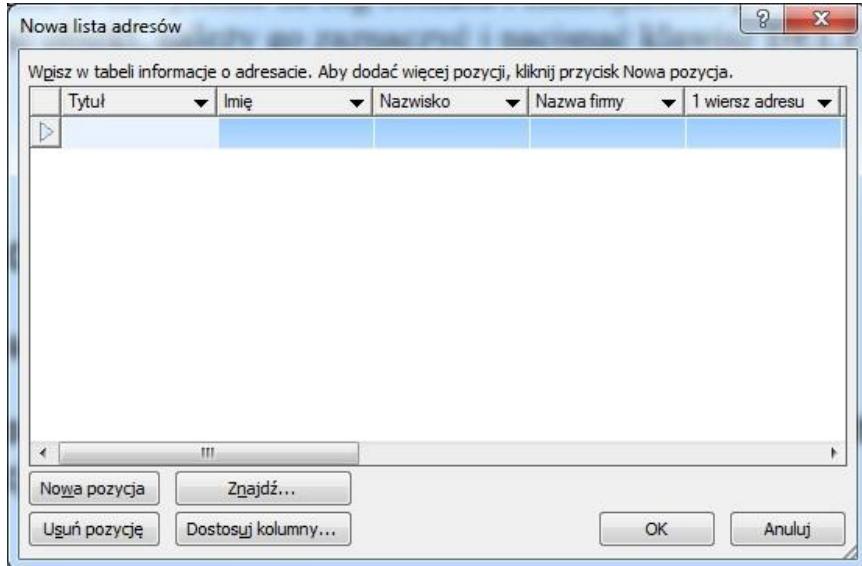


Następnie musimy wpisać adresatów korespondencji lub wybrać istniejący plik z takimi osobami używając klawisza **Wybierz adresatów**, a dalej **Wpisz nową listę**.

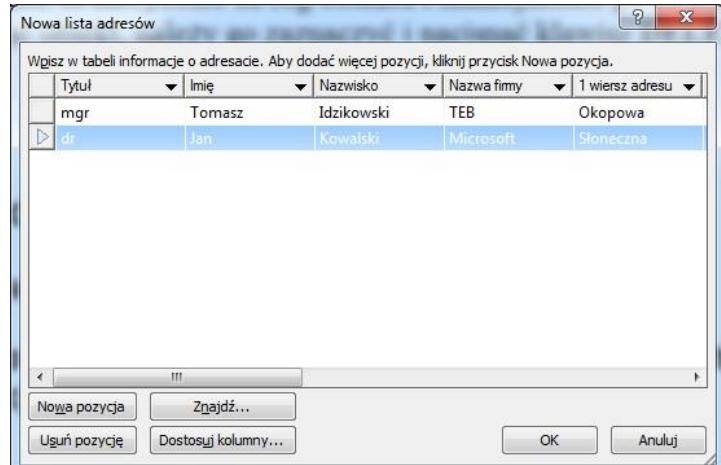




Wpiswanie adresatów odbędzie się w widocznym poniżej okienku



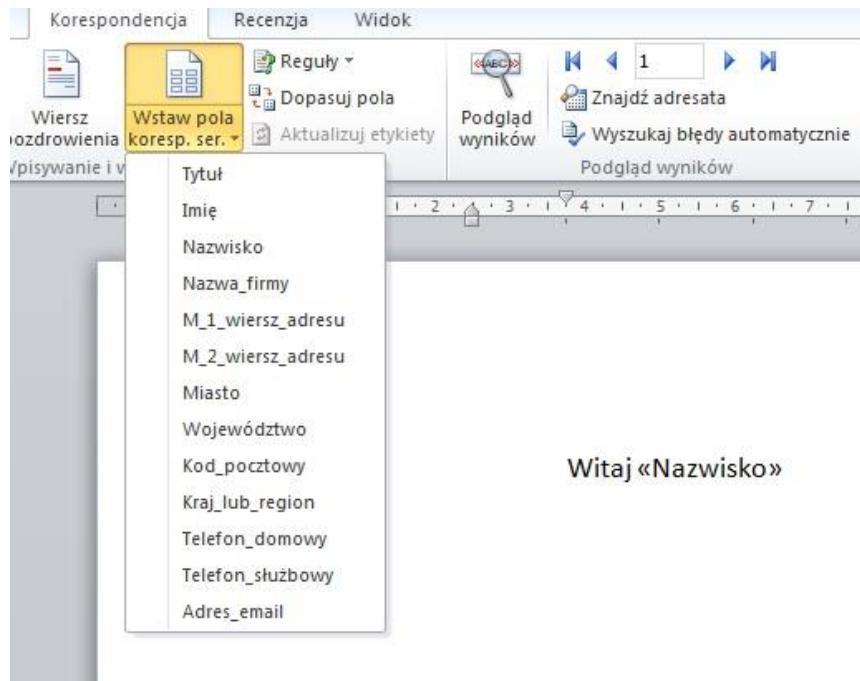
Uzupełniamy listę adresatów.



Naciskamy OK i powstały w ten sposób plik zapisujemy na dysku jako **źródło danych**. Cofamy się do klawisza **Wybierz adresatów** i wybieramy użyci istniejącej listy. Otwieramy źródło danych, które przed chwilą zapisaliśmy.

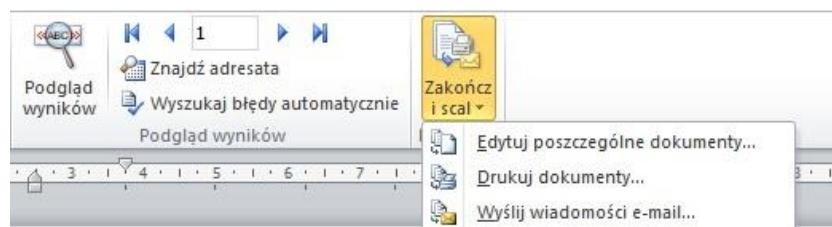
Wracamy do dokumentu głównego i **wstawiamy pola korespondencji seryjnej**. Dostępne będą te pola (tytuł, imię, nazwisko, ...), które uzupełnialiśmy w poprzednim punkcie.

W tym celu wybieramy z paska korespondencji klawisz **Wstaw pole korespondencji seryjnej**. Będą one oznaczone nawiasami <> >.



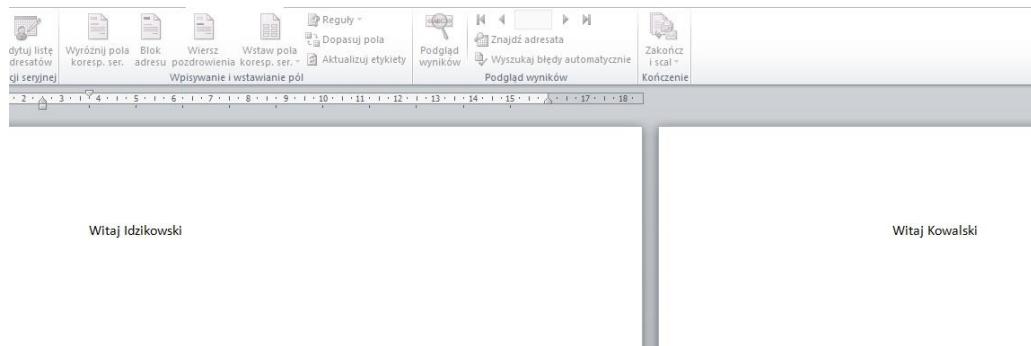
Wydruki

Po utworzeniu dokumentu głównego i uzupełnieniu listy adresowej pozostaje nam scalić oba dokumenty i wydrukować.



W tym celu z paska korespondencji seryjnej wybieramy klawisz **Zakończ i scal**.

Edytuj poszczególne dokumenty oznacza **utworzenie dokumentu wynikowego** w postaci widocznej na poniższym obrazku.

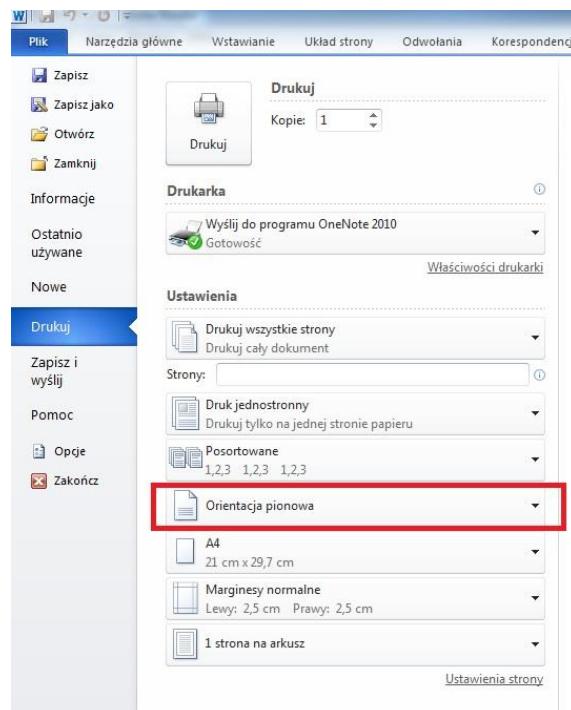


Widzimy, że powstały dwa listy. Każdy do innej osoby.

Drukuj dokumenty na pasku korespondencji seryjnej (po wybraniu Zakończ i scal) oznacza **wydruk gotowego dokumentu**, który widzimy na obrazku powyżej.

Przygotowanie wydruków

Aby zmienić orientację dokumentu należy z menu Plik wybrać **Drukuj** a następnie wybrać Orientację.

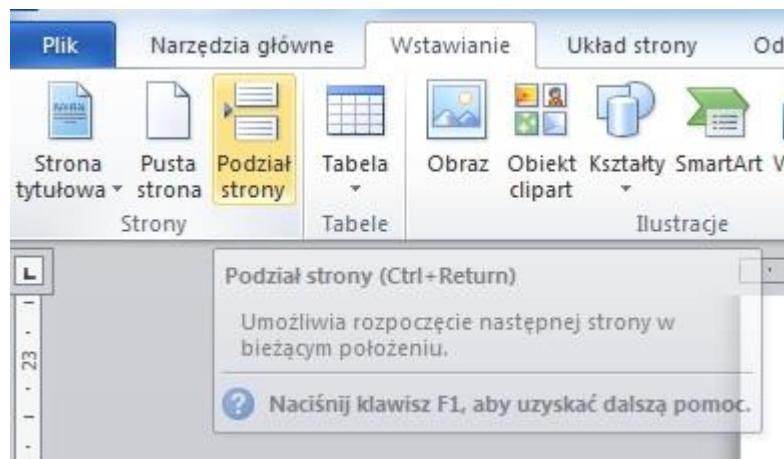


Aby zmienić rozmiar papieru należy użyć opcji znajdującej się poniżej Orientacji.

Aby zmienić marginesy całego dokumentu należy z powyższego obrazka wybrać **Marginesy** a następnie postępować według instrukcji.

Dobrą praktyką w trakcie pisania dłuższego dokumentu jest używanie opcji **Podział strony** zamiast klawisza **ENTER**.

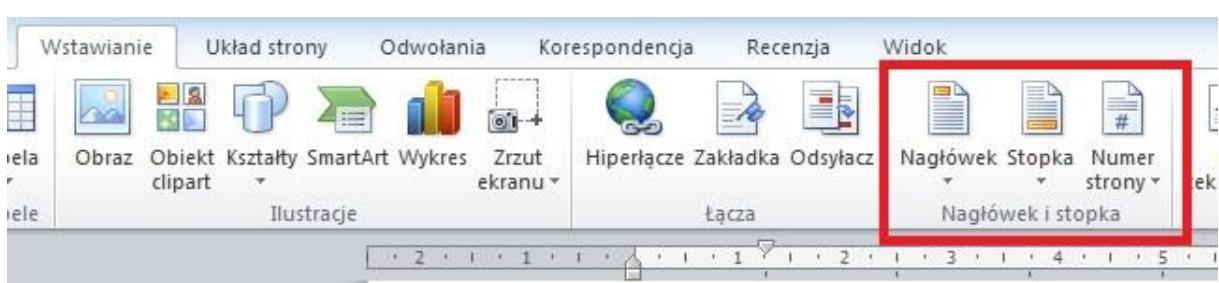
Aby wstawić **Podział strony** należy użyć paska Wstawianie i wybrać Podział strony.



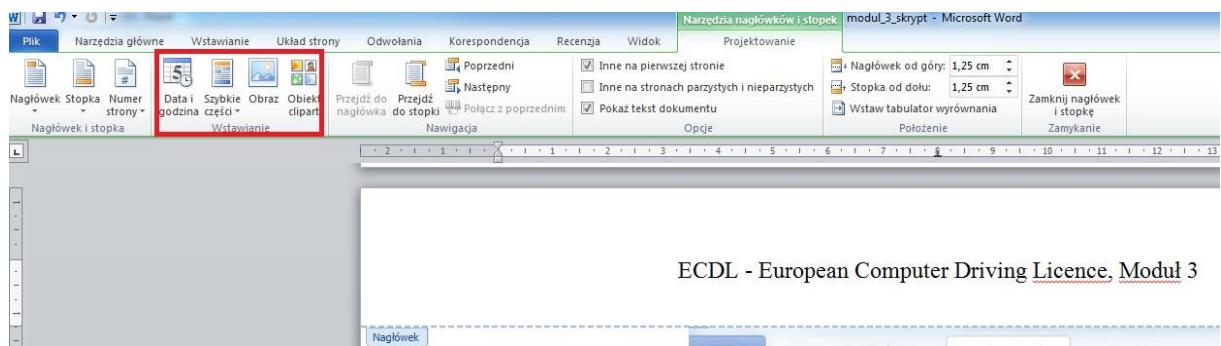
W celu **usunięcia znaku Podziału strony** należy włączyć pokazywanie ukrytych symboli formatowania (**pokaż wszystko**) , następnie skasować znak Podziału strony.

Stopka pozwala nam wstawić dodatkowy tekst na każdej stronie dokumentu **w dolnej części strony**, zaś **nagłówek w górnej części strony**.

Aby wstawić **tekst do stopki lub nagłówka** należy użyć paska **Wstawianie** i wybrać **Nagłówek i stopka**, dalej postępować według instrukcji.



Kiedy po naciśnięciu przycisku **Nagłówek** lub przycisku **Stopka** wybierzemy **Edytuj**, pokaże się nam okienko podobne do poniższego obrazka.



Tutaj możemy **dodawać pola** do nagłówka i stopki. Widoczne pola widać wewnątrz czerwonego zaznaczenia. Można więc dodać **Datę i Godzinę**, **Obraz**, a po wybraniu **Szybkie części** można dodać **nazwę pliku**. Obok widzimy również, że mamy możliwość dodania **numeru strony**.

Aby wstawić do dokumentu automatyczne numerowanie stron, należy wybrać Numer strony, tak jak na poniższym obrazku.

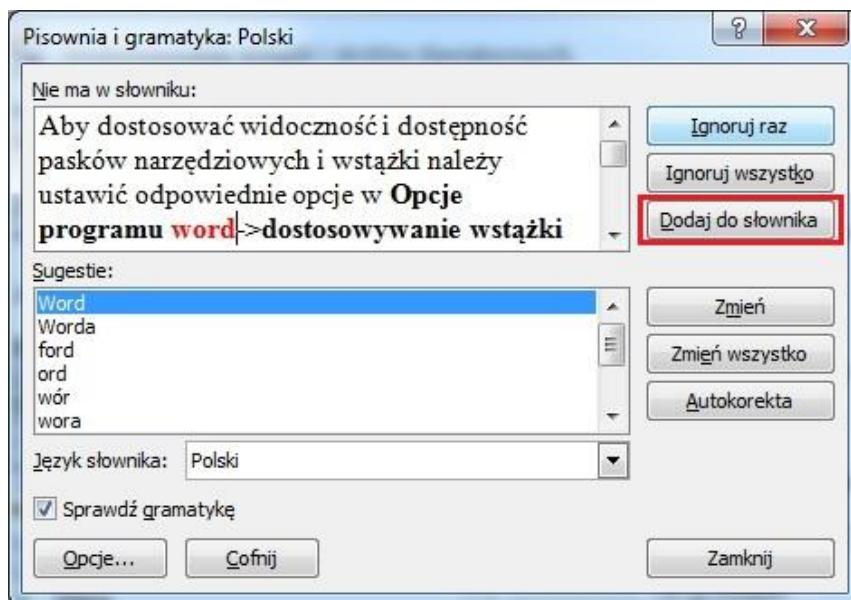


Po zakończeniu wprowadzania tekstu do edytora tekstu, dokument **należy wydrukować**. Ale zanim to zrobimy, to musimy **sprawdzić czy nie ma błędów** i je ewentualnie **poprawić**.

Aby **sprawdzić i poprawić błędy** w dokumencie należy użyć **Pisownia i gramatyka** z paska Recenzja->Sprawdzanie.

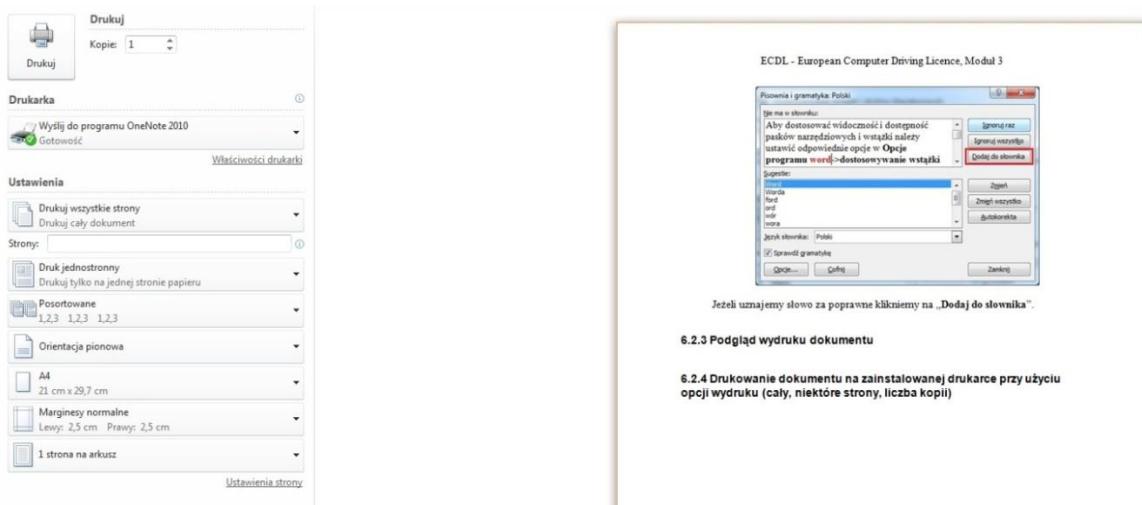


W trakcie sprawdzania pisowni może się okazać, że **chcemy dodać nowe słowo do słownika**.

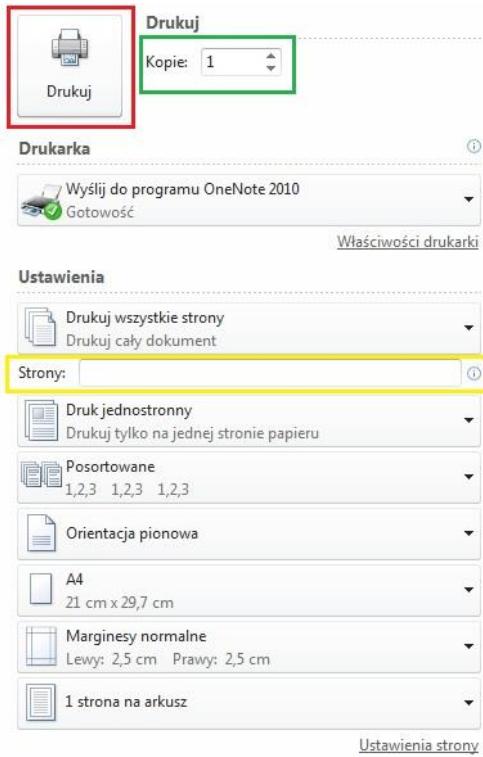


Jeżeli uznajemy słowo za poprawne klikniemy na „**Dodaj do słownika**”.

Aby sprawdzić wygląd dokumentu przed wydrukiem należy z paska Plik wybrać Drukuj. Wtedy zobaczymy okienko podobne do poniższego obrazka.



Z lewej strony mamy ustawienia wydruku, a z prawej widzimy **podgląd wydruku dokumentu**.



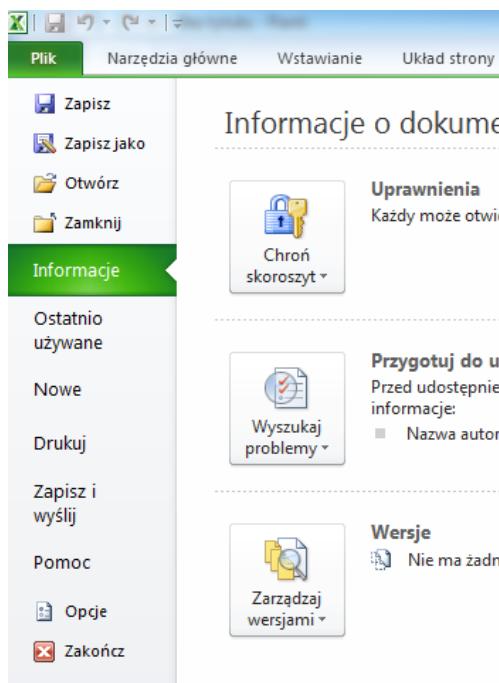
Aby **wydrukować cały dokument** należy nacisnąć przycisk **Drukuj**. Jeśli chcemy **wydrukować kilka kopii** należy wprowadzić odpowiednią liczbę w polu **Kopie**. Jeżeli zależy nam **tylko na kilku stronach**, to w polu **Strony** należy podać ich numery.

Arkusz kalkulacyjny MS Excel

Podstawowe informacje

Aby **uruchomić** arkusz kalkulacyjny należy wybrać odpowiednią ikonę z menu start o nazwie Microsoft Excel. Do uruchomienia arkusza kalkulacyjnego Microsoft Excel wystarczy również otwarcie pliku wcześniej w nim zapisanego.

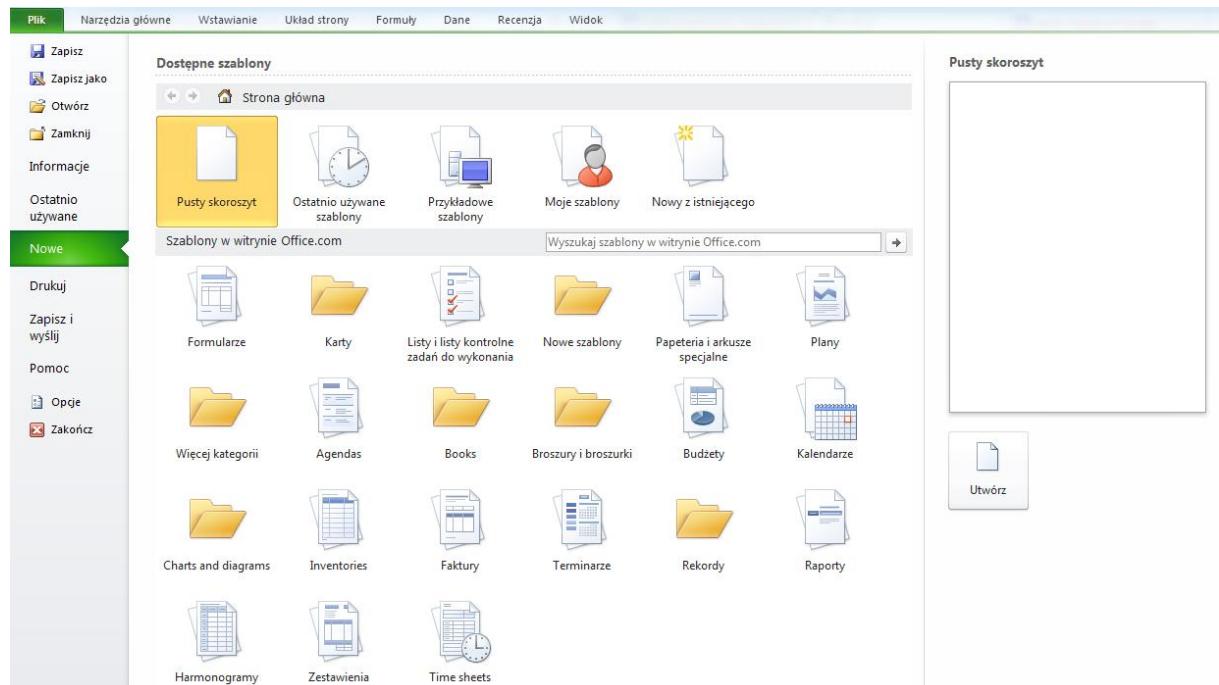
Aby **zakończyć** pracę z arkuszem kalkulacyjnym Microsoft Excel należy z menu górnego Plik wybrać opcję **zakończ**.



Aby **otworzyć dokument** (skoroszyt składający się z wielu arkuszy) należy wybrać opcję **Otwórz** z widocznego na powyższym obrazku menu Plik.

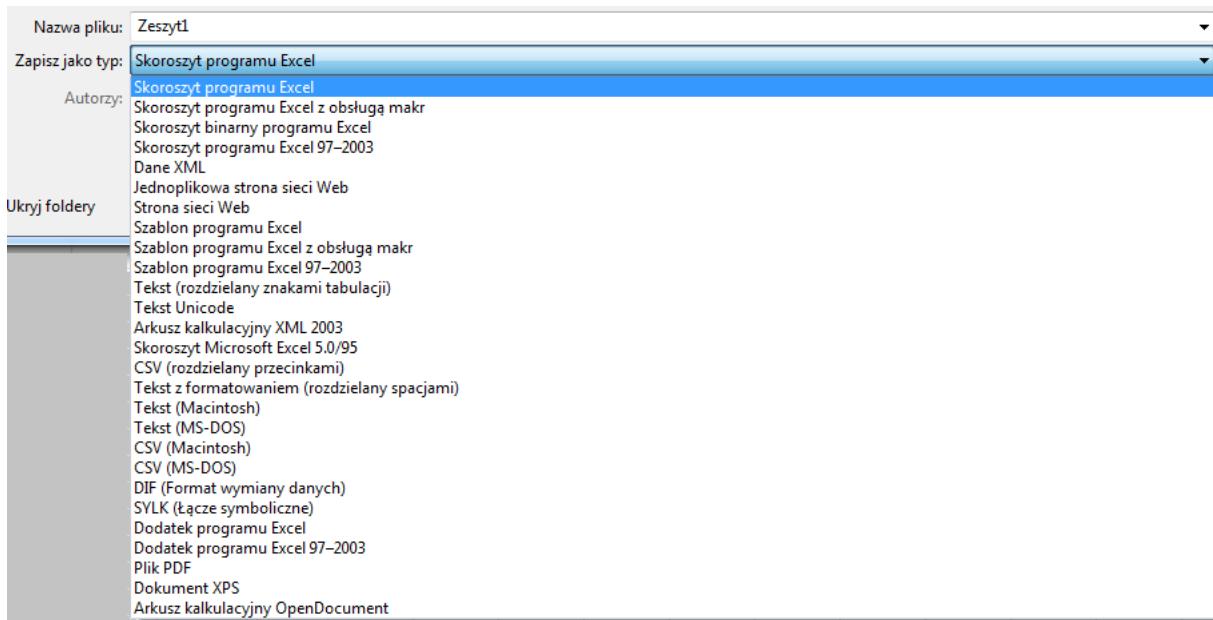
Aby **zakończyć pracę z dokumentem** należy wybrać opcję **Zamknij**.

Aby utworzyć dokument w oparciu o **domyślny szablon** należy z menu plik wybrać opcję **nowe**. Na poniższym obrazku widać dostępne szablony.



Aby **zapisać plik** na dysku z domyślną lub pod określoną nazwą należy użyć opcji **zapisz jako** z menu Plik.

Po wybraniu opcji zapisz jako z menu plik i po rozwinięciu **zapisz jako typ** pojawia się nam możliwość zapisania naszego dokumentu jako pliku innego typu.

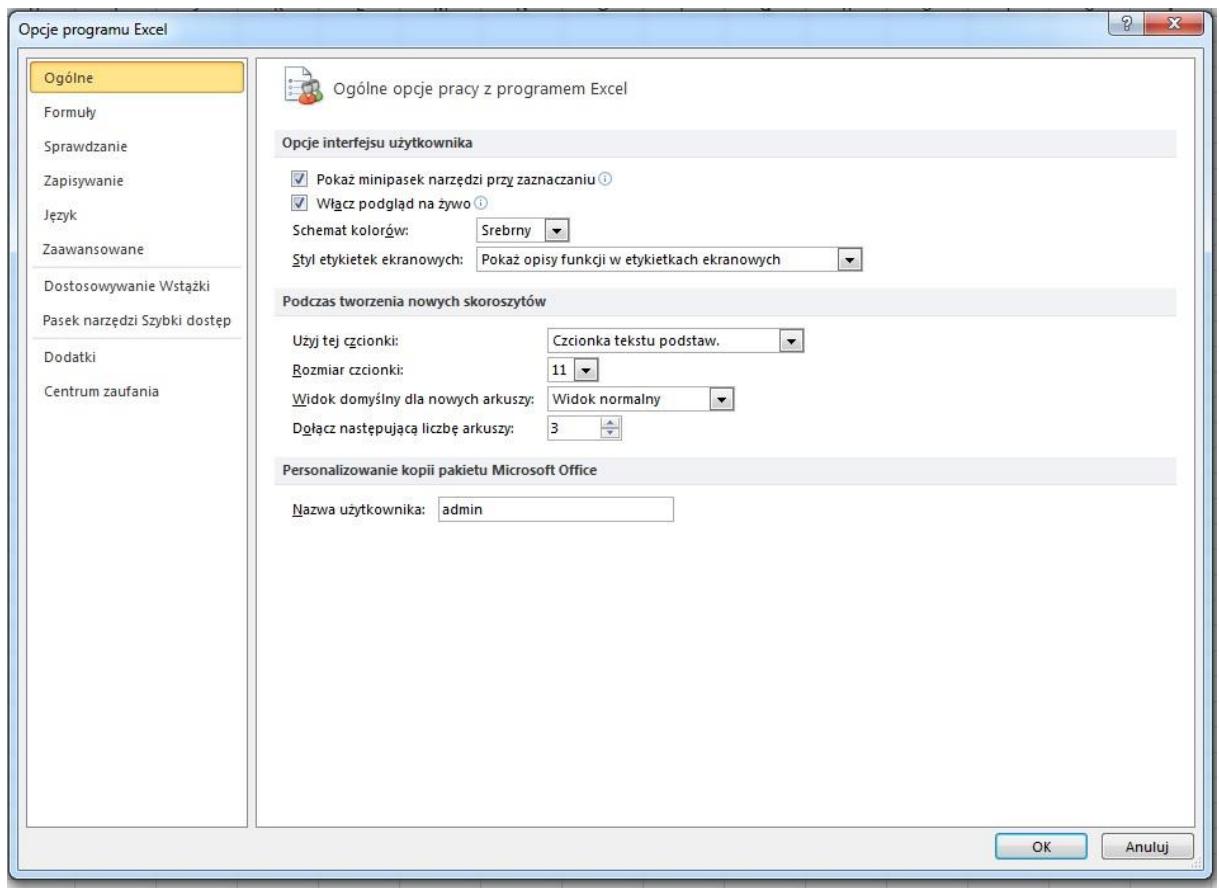


Microsoft Excel otwiera każdy dokument w innym oknie. Do poruszania się między **otwartymi dokumentami** służy więc kombinacja klawiszy **ALT-TAB** lub przyciski maksymalizacji i minimalizacji okna.

Skoroszyt składa się z jednego lub wielu arkuszy. Aby w obrębie jednego dokumentu (składającego się z co najmniej dwóch arkuszy) **przełączać się między arkuszami** należy wybrać z dolnego paska odpowiedni arkusz.



Do ustawienia podstawowych preferencji dla arkusza kalkulacyjnego służy aplet **opcje programu Excel**. Aby go uruchomić należy z menu plik wybrać **Opcje**. Wtedy zobaczymy

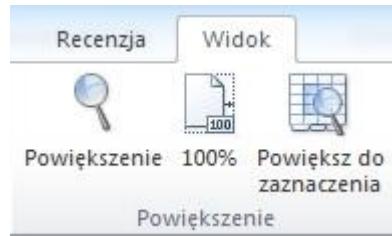


Zauważmy od razu, że możemy tutaj wybrać **domyślną nazwę użytkownika** dla tworzonych dokumentów („Ogólne->Personalizowanie kopii pakietu Microsoft Office->nazwa użytkownika”).

Podobnie **domyślny folder zapisu naszych dokumentów** znajdziemy w „Zapisywanie->Domyślna lokalizacja pliku”.

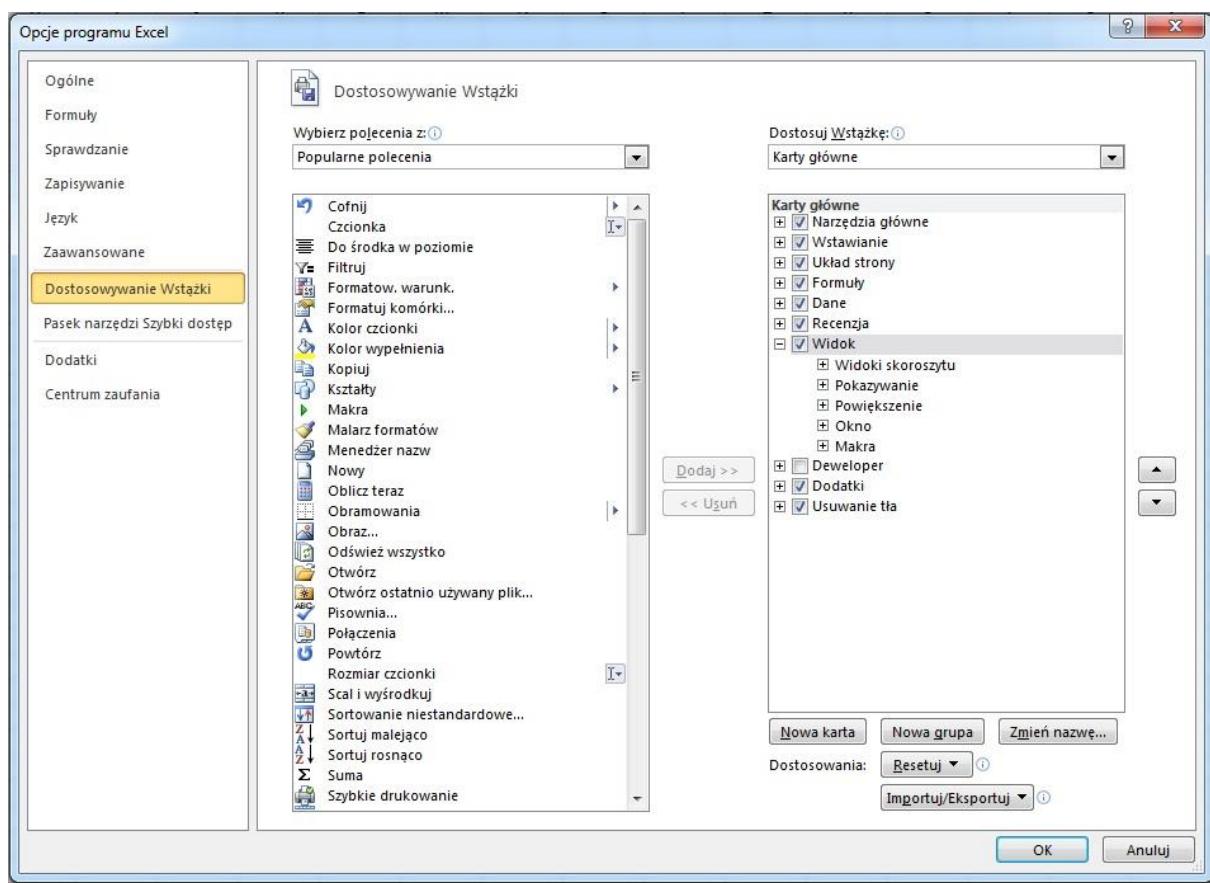
Informacje na temat pomocy znajdziemy wybierając opcję **pomoc** z menu Plik.

Aby **powiększyć wyświetlanie dokumentu** należy wybrać z menu górnego widok. Wtedy zobaczymy



Drugi sposób – szybszy – powiększania widoku to naciśnięcie klawisza **CTRL** i zastosowanie **scrolla myszki**.

Aby dostosować widoczność i dostępność pasków narzędziowych i wstążki należy ustawić odpowiednie opcje w **Opcje programu Excel->dostosowywanie wstążki**



Komórki

Komórka jest nośnikiem pojedynczej informacji. Każdą komórkę można zlokalizować podając jej **adres**. Komórka składa się z **nazwy kolumny** i **numeru wiersza**. Na obrazku poniżej widać pojedynczą komórkę o adresie **A1**.

Czcionki		
Schowek	A1	▼
	A	C
1	A1	B
2		
3		
4		

Do **dobrych praktyk** w przypadku tworzenia list używając arkusza kalkulacyjnego należą:

- unikanie pustych wierszy i kolumn w głównej części listy,
- wstawianie pustych wierszy w miejscu poprzedzającym podsumowanie wierszy,
- upewnienie się, że komórki obramowujące listę są puste

Do każdej komórki można wprowadzić: **liczby, daty** oraz **tekst**. Dane w komórce muszą być tego samego typu. Dane do komórki wprowadzamy w następujący sposób: Najpierw wybieramy interesującą nas komórkę, następnie klikamy szybko dwukrotnie lewym klawiszem myszy. **Teraz możemy wprowadzić dane.**

Aby zaznaczyć **pojedynczą komórkę** należy na nią kliknąć myszką.

Aby zaznaczyć **blok dowolnych komórek** należy zaznaczyć pierwszą z komórek a następnie nie puszczaając lewego klawisza myszy wskazać ostatnią komórkę. Drugi sposób to **użycie pola nazwy**. Na obrazku poniżej widać zaznaczone komórki od B2 do D4.

Czcionka			
Schowek		Czcionka	
B2:D4		f _x	
A	B	C	D
1			
2			
3			
4			
5			

Aby zaznaczyć **cały arkusz** należy wybrać kombinację klawiszy **CTRL-A**.

Aby **edytować komórkę** wystarczy na nią kliknąć dwukrotnie lewym klawiszem myszy.

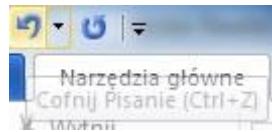
Czcionka					
Schowek	Czcionka				
E2	dwa				
A	B	C	D	E	F
1					
2				dwa	
3					
4					
5					

Edycję możemy wtedy przeprowadzić na dwa sposoby.

Założymy, że mamy **blok komórek** wypełnionych wartościami 1. Chcemy aby zamiast tych wartości pojawiły się tam liczby 2. Aby to zrobić, należy do pierwszej komórki wpisać liczbę 2, zaznaczyć komórkę z wartością 2 i **przeciągnąć ją** na pozostały obszar, tak jak na poniższym rysunku.

2	1	1	1
2	1	1	1
2	1	1	1
2	1	1	1
2	1	1	1
2	1	1	1
2	1	1	1
2	1	1	1
2	1	1	1
2	1	1	1

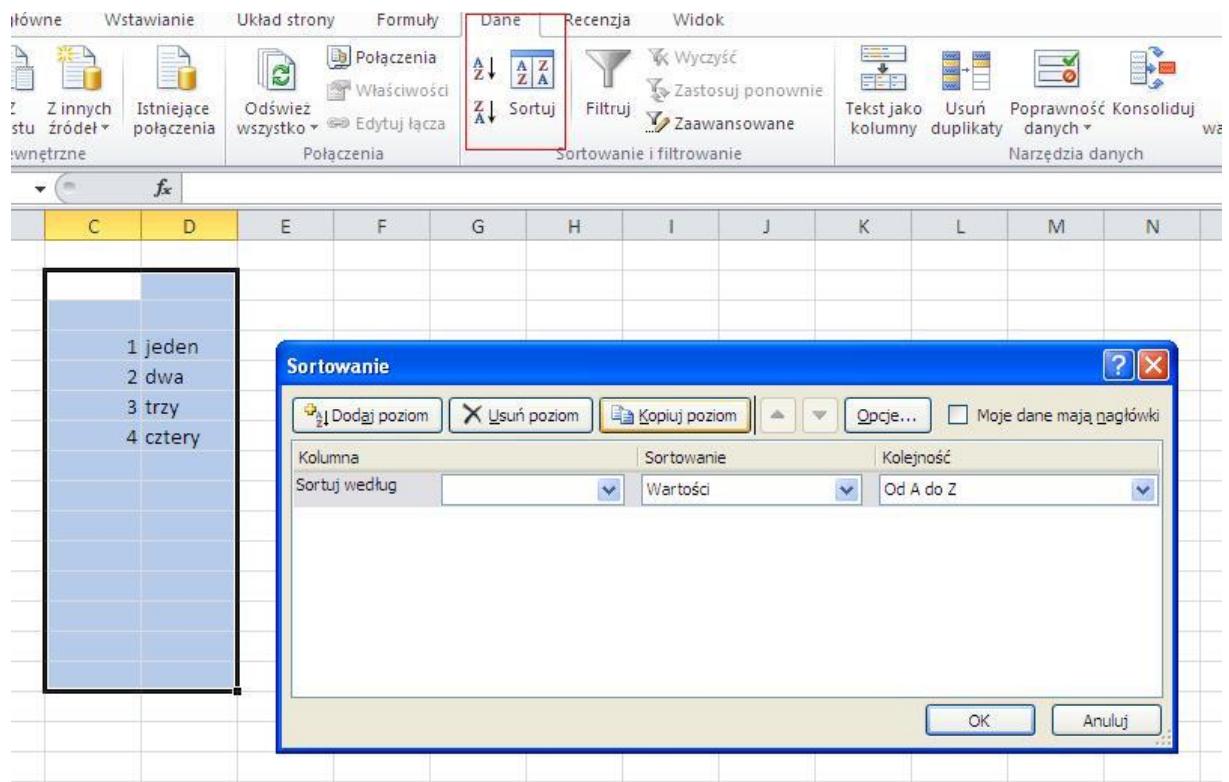
Aby **Cofnąć** poprzednią operację albo wykonać **Ponów** należy skorzystać z paska szybkiego dostępu.



Aby wyszukać określoną wartość lub tekst w arkuszu należy wybrać **Narzędzia główne->Edytowanie->Znajdź i Zaznacz->Znajdź** lub nacisnąć kombinację klawiszy **CTRL-F**.

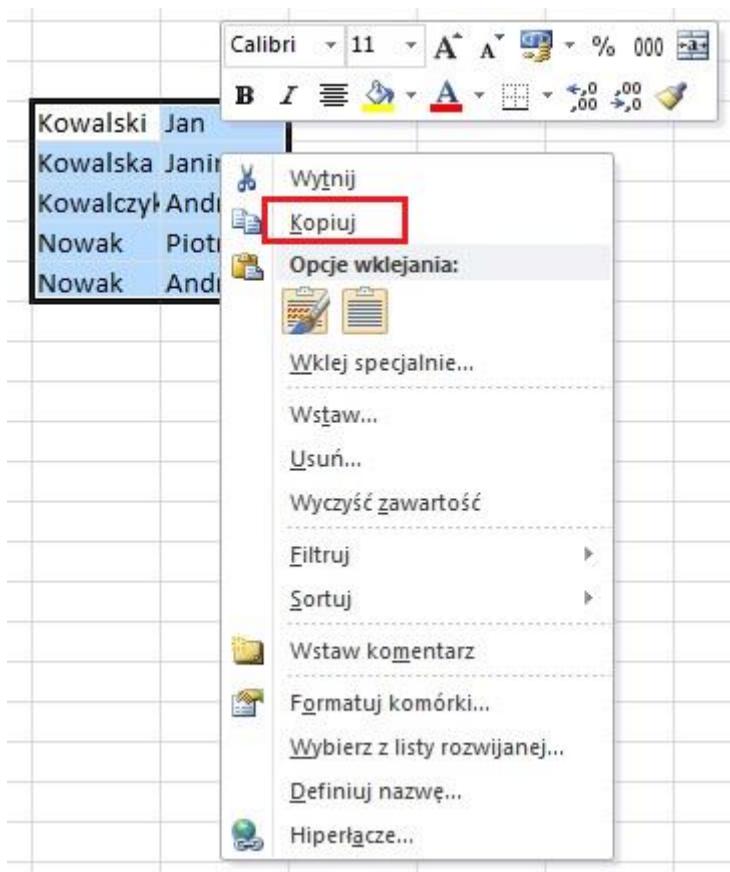
Aby zmienić określoną wartość lub tekst w arkuszu należy wybrać **Narzędzia główne->Edytowanie->Znajdź i Zaznacz->Zamień** lub nacisnąć kombinację klawiszy **CTRL-H**.

Aby **posortować dane** należy zaznaczyć obszar, a następnie wybrać z paska narzędzi **Dane->Sortowanie i filtrowanie->Sortuj**. Tutaj możemy posortować dane według **określonej kolumny** w porządku **rosnącym** lub **malejącym**.



Kopiowanie zawartości

Zaznaczamy komórkę lub blok komórek które zamierzamy skopiować (patrz 2.1.4.) Następnie klikając prawym klawiszem myszy otwieramy menu kontekstowe i wybieramy **kopiuj** lub wciskamy kombinację klawiszy **CTRL+C**.



Następnie zaznaczamy miejsce docelowe, otwieramy prawym klawiszem myszki menu kontekstowe i wybieramy **Wklej** lub wciskamy kombinację klawiszy **CTRL-V**.

Automatyczne wypełnianie komórek

Automatyczne wypełnianie komórek danymi przydaje się nam wtedy kiedy wartości w kolejnych komórkach różnią się o stałą wartość.

1	
2	
	4

Na powyższym obrazku widać działanie automatycznego wypełniania komórek w przypadku dwóch pierwszych komórek wypełnionych liczbami 1 oraz 2. Aby zrealizować to zadanie należy

zaznaczyć dwie pierwsze komórki myszką. Wtedy w prawym dolnym rogu zaznaczenia pojawi się nam czarny krzyżyk. Należy go chwycić i przeciągnąć w dół. Otrzymamy wtedy kolejne liczby, a więc 3,4.

Aby przenieść zawartość komórki należy zaznaczyć daną komórką, następnie z menu kontekstowego wybrać Wytnij, a w miejscu docelowym wybrać Wklej.

UWAGA:

Tego sposobu przenoszenia nie stosujemy w przypadku komórek zawierających formuły.

Aby usunąć zawartość komórki wystarczy ją zaznaczyć i nacisnąć klawisz DELETE lub wybrać w menu kontekstowym Usuń.

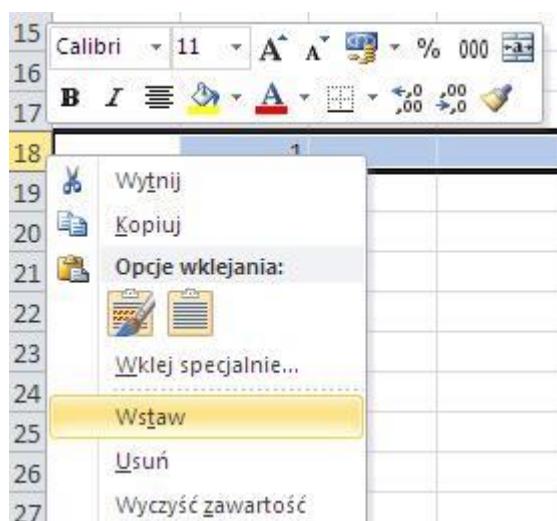
Zarządzanie arkuszami

Aby zaznaczyć wiersz należy kliknąć na jego numer lewym klawiszem myszy.

Jeśli chcemy zaznaczyć kilka sąsiednich wierszy to musimy kliknąć pierwszy z nich trzymając lewy klawisz myszki, a następnie przesunąć kurSOR myszki do ostatniego wybranego wiersza. Jeśli zrobimy to z użyciem klawisza CTRL to zaznaczymy wybrane wiersze.

Aby zaznaczyć kolumnę należy kliknąć na jej nazwie (literce) lewym klawiszem myszy. Jeśli chcemy zaznaczyć kilka sąsiednich kolumn to musimy kliknąć pierwszą z nich trzymając lewy klawisz myszki, a następnie przesunąć kurSOR myszki do ostatniej wybranej kolumny. Jeśli zrobimy to z użyciem klawisza CTRL to zaznaczymy wybrane kolumny.

Jeśli chcemy wstawić wiersz należy najpierw zaznaczyć wiersz, następnie z menu kontekstowego wybrać Wstaw. Podobnie robimy w przypadku kolumny. Aby usunąć wiersz lub kolumnę należy najpierw zaznaczyć obiekt i menu kontekstowe wybrać Usuń.



Aby zmienić **szerokość kolumn** lub **wysokość wierszy** należy najpierw zaznaczyć obiekty, które mają podlegać tej zmianie. Następnie z menu kontekstowego należy wybrać szerokość kolumny lub wysokość wiersza odpowiednio.

Aby **zamrozić (zablokować)** wiersz, należy zaznaczyć **wiersz następny** a następnie z paska **Widok->Okno** wybrać klawisz **Podziel**. Wtedy zablokowany wiersz będzie widoczny nawet gdy znajdziemy się kilkaset wierszy niżej. Podobnie postępujemy w przypadku kolumny. Aby odmrozić wiersz lub kolumnę należy odznaczyć **Podziel**.

Arkusze

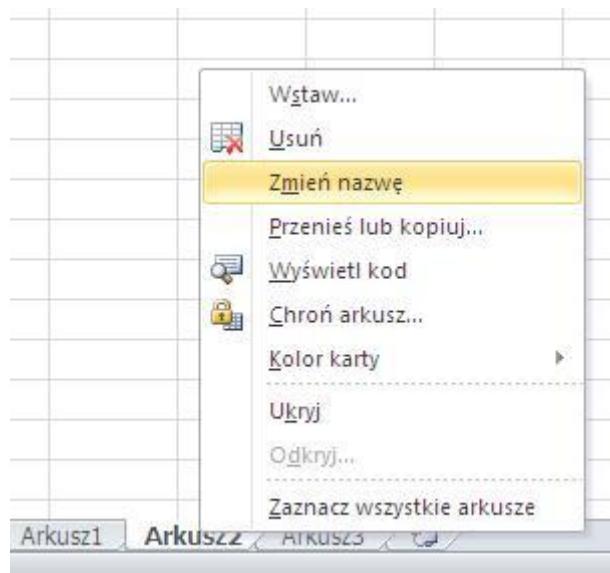
Aby **zamienić miejscami arkusze** należy kliknąć na jeden z nich i przeciągnąć go w nowe miejsce. Operacja jest zilustrowana na poniższym obrazku.



Aby **wstawić nowy arkusz** należy wywołać menu kontekstowe dla jednego z arkuszy, których nazwy widzimy w dolnej części okienka. Z dostępnych opcji wybrać wtedy **Wstaw**. Jeśli chcemy **usunąć dany arkusz**, to wybierzemy **Usuń**.

Wskazane jest **stosowanie dobrych praktyk** w nadawaniu nazw arkuszom. Do dobrych praktyk zaliczymy **stosowanie nazw**, które niosą **informacje co się w danym arkuszu znajduje**.

Aby **skopiować arkusz** należy z menu kontekstowego wybrać **Skopiuj**. Jeśli chcemy tylko **zmienić nazwę** to wybierzemy **Zmień nazwę**.



Reguły i funkcje

Arkusa kalkulacyjnego nie należy traktować jak większego kalkulatora. Ma on o wiele potężniejsze możliwości niż kalkulator. Kolejną rzeczą, z której należy sobie zdać sprawę jest fakt, że w arkuszu należy używać dostępnych formuł w celu wykonywania operacji arytmetycznych. Jeśli chcemy dodać do siebie dwie liczby to po wpisaniu ich w odpowiednie komórki należy użyć formuły **SUMA**. W takiej formule nie będziemy operowali liczbami, ale adresami komórek, do których te liczby wpisaliśmy.

Aby utworzyć regułę należy ustawić cursor w wybranej komórce (tam gdzie chcemy uzyskać wynik), napisać **=**, a następnie napisać pożądaną regułę. Na obrazku widzimy regułę, która doda do siebie liczby zapisane w trzech komórkach. Po wpisaniu reguły należy nacisnąć **ENTER**. Powinniśmy uzyskać wynik 6.

SUMA				
A	B	C	D	E
		1 =B2+B3+B4		
	2			
	3			

Błąd **#NAZWA?** występuje wtedy gdy wpisujemy do komórki coś innego niż program Excel by się spodziewał, np. w przypadku formuły sumowania zamiast adresów komórek lub liczb wpiszemy jakiś tekst.

Dzialania				
A	B	C	D	E
	#NAZWA?			

Błąd **#DZIEL/0!** oznacza dzielenie przez zero. Występuje wtedy gdy próbujemy podzielić dowolną liczbę przez komórkę, w której znajduje się liczba 0 lub przez pustą komórkę.

Schowek	Czcionka			
B3	f _x =B2/C2			
A	B	C	D	
		3	0	
	④ #DZIEL/0!			

Błąd **#ADR!** oznacza odwołanie do nieistniejącego adresu komórki. Wystąpi wtedy, gdy będziemy chcieli użyć adresu komórki, który nie istnieje. Błąd ten może wystąpić w przypadku kopiowania reguł w których zastosowaliśmy adresowanie względne.

Adresowanie

Adresowanie bezwzględne to odwołanie do konkretnej komórki o konkretnym adresie. Taki adres komórki występujący w formule i skopiowany do innej komórki nie ulega zmianie. W tym typie adresowania należy przed nazwą kolumny i numerem wiersza dodać znak \$.

B9	f _x	=ILOCZYN(B2*\$C\$2)		
A	B	C	D	E
	2	3		
	1			
	2			
	3			
	4			
		6		
		3		

Adresowanie względne to taki sposób adresowania komórek, w wyniku którego zapamiętywane jest położenie komórki, do której odwołuje się formuła względem komórki, w której znajduje się ta formuła. Następnie w wyniku skopiowania formuły zawierającej komórki zaadresowane względnie pozostaje taka sama, mimo że zmianie podlegają adresy komórek.

B9		f(x)	=ILOCZYN(B2*C2)	
A	B	C	D	E
	2	3		
	1			
	2			
	3			
	4			
	6			
	0			

Funkcje

Aby **zastosować funkcję** w wybranej komórce arkusza należy najpierw ustawić cursor w wybranej komórce (tam gdzie chcemy uzyskać wynik), napisać **=**, a następnie napisać pożdaną funkcję.

Do **najczęściej stosowanych funkcji** używanych w arkuszu Excel należą:

- | | |
|---------------------------------|-----------------------|
| • sumowanie | suma |
| • obliczanie średniej | średnia |
| • wyznaczanie minimum | min |
| • wyznaczanie maksimum | max |
| • zliczanie | licz.jeżeli |
| • obliczanie niepustych komórek | ile.niepustych |
| • zaokrągalanie | zaokr |

Na poniższym rysunku widzimy sposób użycia formuły **licz.jeżeli** do sprawdzenia ile razy wystąpiła liczba 1 w podanym zakresie. Po wpisaniu formuły okazuje się, że wynikiem jest liczba 4.

SUMA	X	✓	fx	=LICZ.JEŻELI(B2:B10)
A	B	C	D	E
	2			
	3			
	1			
	1			
	2			
	1			
	2			
	4			
	1			
		=LICZ.JEŻELI(B2:B10)		

Bardzo często będziemy stosowali funkcję **JEŻELI**. Jej składnię można przedstawić w następujący sposób:

=JEŻELI(warunek;co robić jeśli prawdziwy; co robić jeśli nie jest prawdziwy)

Służy ona do **sprawdzania spełnienia warunku** w określonej komórce i działania w zależności od jego spełnienia lub nie. Adres komórki podlegającej sprawdzeniu podajemy do funkcji jako pierwszy parametr (**warunek**). Następnie musimy określić co zrobić jeśli warunek jest prawdziwy (**co robić jeśli prawdziwy**) oraz po średniku co zrobić jeśli warunek nie jest spełniony (**co robić jeśli nie jest prawdziwy**).

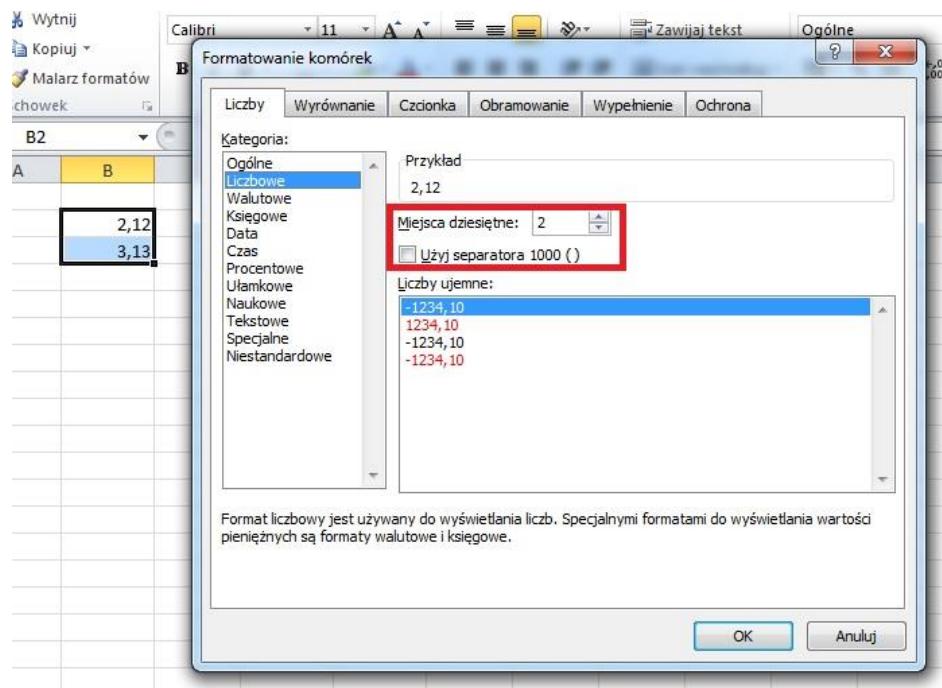
Na obrazku poniżej widzimy przykładowe zastosowanie tej funkcji do sprawdzenia czy w komórce o adresie B2 jest liczba dodatnia (wtedy dostaniemy komunikat TAK) lub ujemna (wtedy dostaniemy komunikat NIE).

SUMA	X	✓	fx	=JEŻELI(B2;"TAK";"NIE")
A	B	C	D	E
	2	=JEŻELI(B2;"TAK";"NIE")		

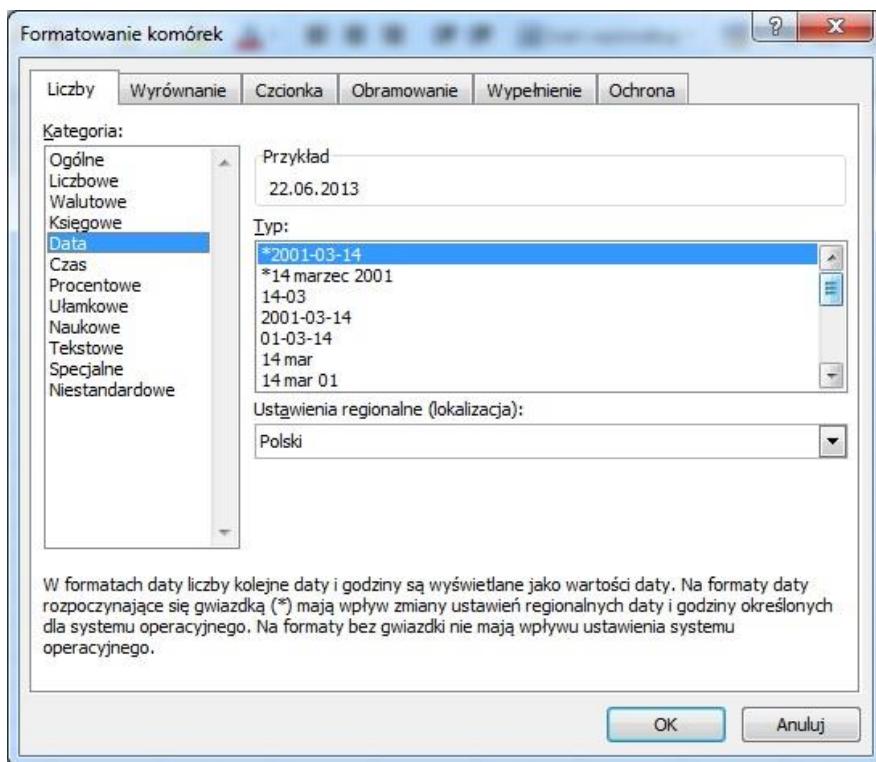
Powinniśmytrzymać w komórce C2 (tam gdzie wpisujemy funkcję) odpowiedź TAK.

Formatowanie

Aby sformatować komórki zawierające liczby należy zaznaczyć te komórki i z menu kontekstowego wybrać **Formatowanie komórek**. Po wybraniu **Liczbowe** w kategorii widzimy opcję zmiany wyświetlanych miejsc dziesiętnych oraz użycia separatora grup tysięcy.

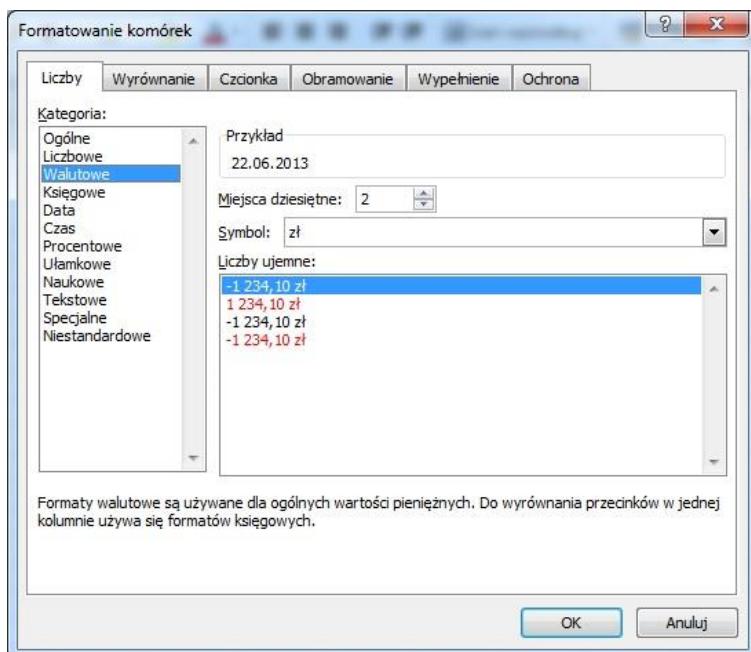


Aby sformatować komórki zawierające daty należy wybrać z **Formatowanie komórek** zakładkę **Data**. Wtedy zobaczymy to, co widać na poniższym rysunku.



Tutaj możemy sobie wybrać **właściwy sposób wyświetlania daty**.

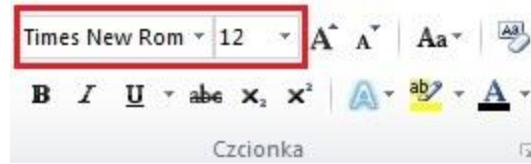
W celu sformatowania komórek zawierających liczby, aby **wyświetlić symbol waluty** należy użyć kategorii **Waluta**.



Aby wyświetlić wartość procentową liczby w komórce należy użyć kategorii **Procentowe**.

Format komórek

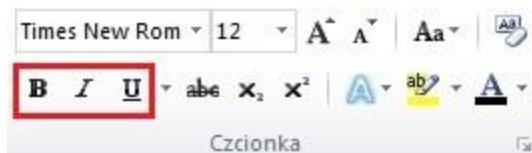
Zmiana formatu tekstu(kroju i wielkości czcionki) jest dostępna na wstążce na pasku Narzędzia główne -> Czcionka.



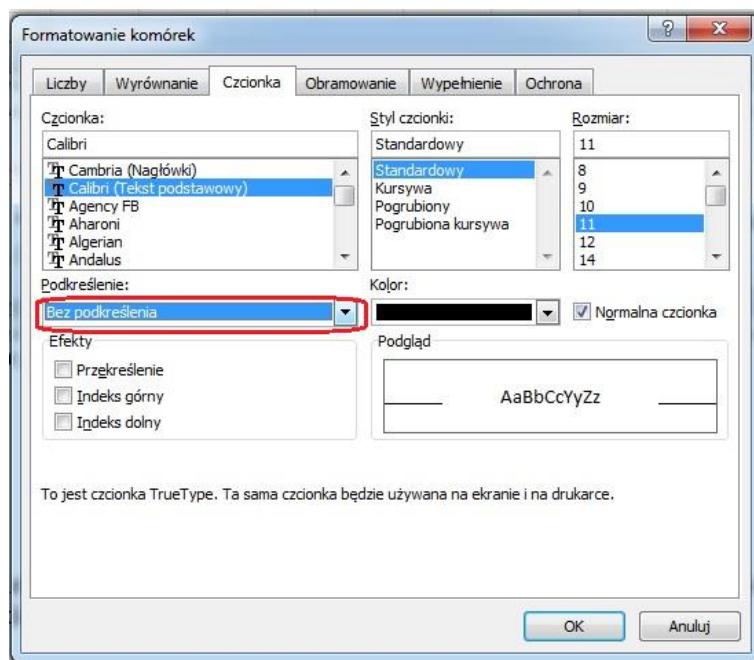
Wielkość czcionki można również zmienić w menu kontekstowym, które otwieramy prawym klawiszem myszki po zaznaczeniu komórki.

Zmianę stylu czcionki dokonujemy w tym samym miejscu co zmianę formatu tekstu.

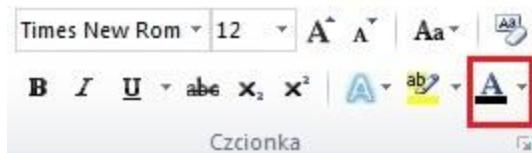
Aby napisać tekst czcionką: **pogrubioną** wybierzemy klawisz **B**, **pochyloną** **I**, a **podkreślona** klawisz **U**.



Aby ustawić podkreślenie możemy również skorzystać z menu kontekstowego i wybrać **Formatuj komórki**, a następnie użyć zakładki **Czcionka**.



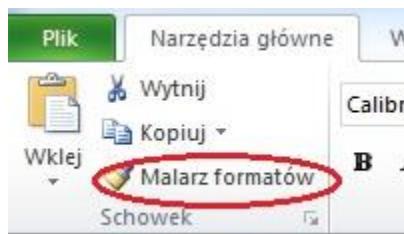
Kolor czcionki można zmienić w miejscu zaznaczonym na czerwono na poniższym obrazku.



Wypełnienie tła znajdziemy w menu kontekstowym, następnie **Formatuj komórki->Wypełnienie**.

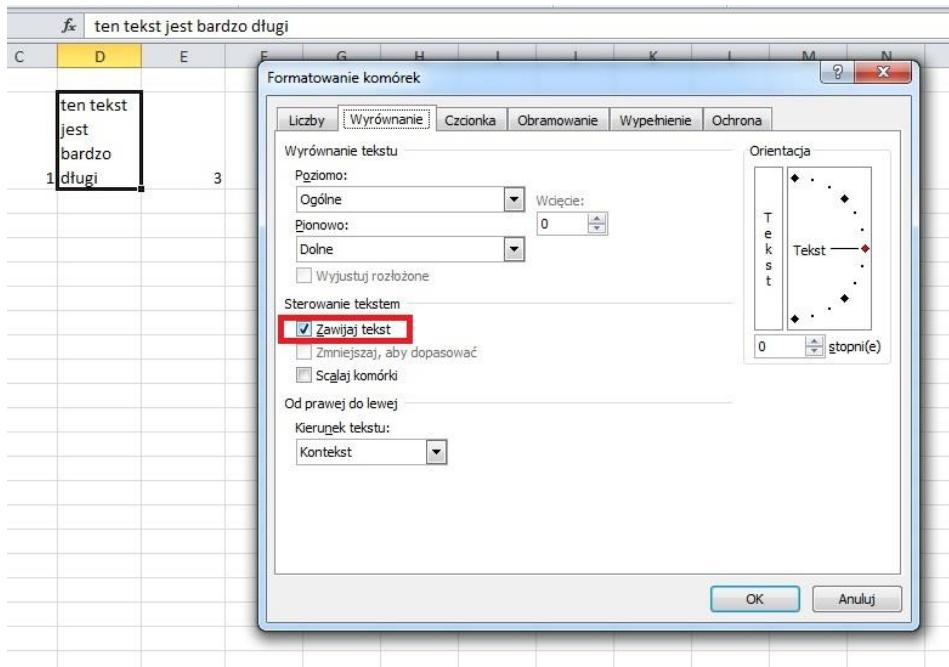
Kopiowanie formatu komórki do innej komórki

Zaznaczamy komórkę zawierającą format który chcemy skopiować w inne miejsce. Następnie klikamy na „**Malarz formatów**” (Narzędzia główne ->Schowek) i klikamy lewym przyciskiem myszy na komórkę (lub blok komórek) gdzie ma być skopiowany format z jednej komórki.

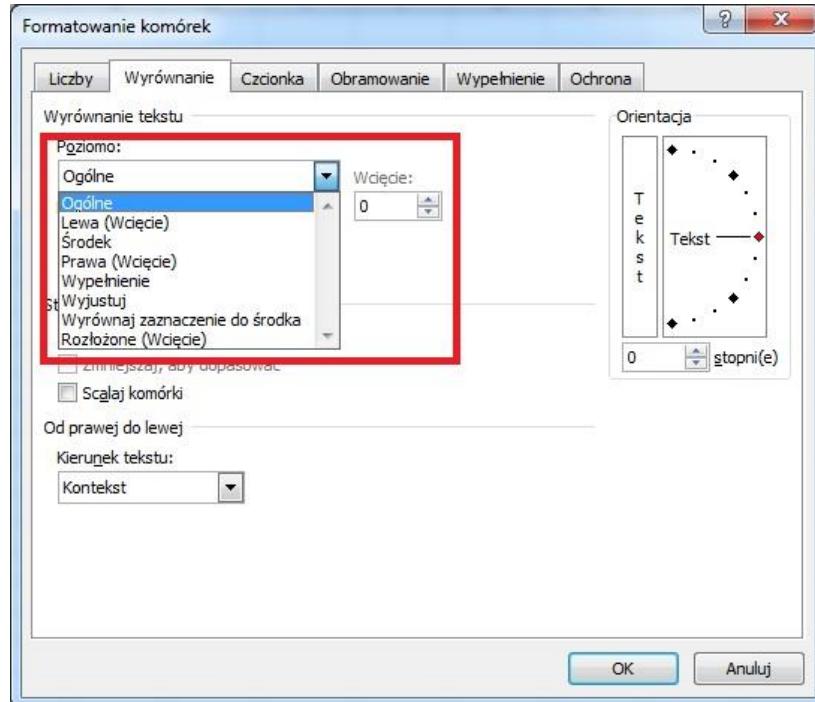


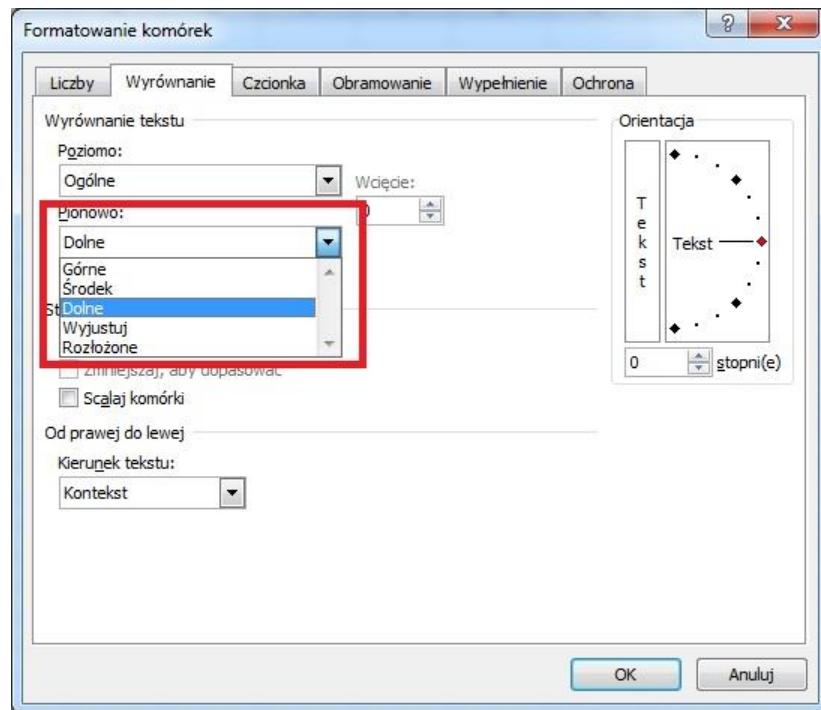
Podobnie robimy w przypadku kopiowania formatu dla bloku komórek.

Czasami zdarza się, że do komórki wpisujemy długi tekst, który później nie jest wyświetlany w całości na ekranie. Aby temu zaradzić i spowodować, aby mimo wszystko cały tekst był wyświetlany na ekranie należy zastosować **zawijanie tekstu**.

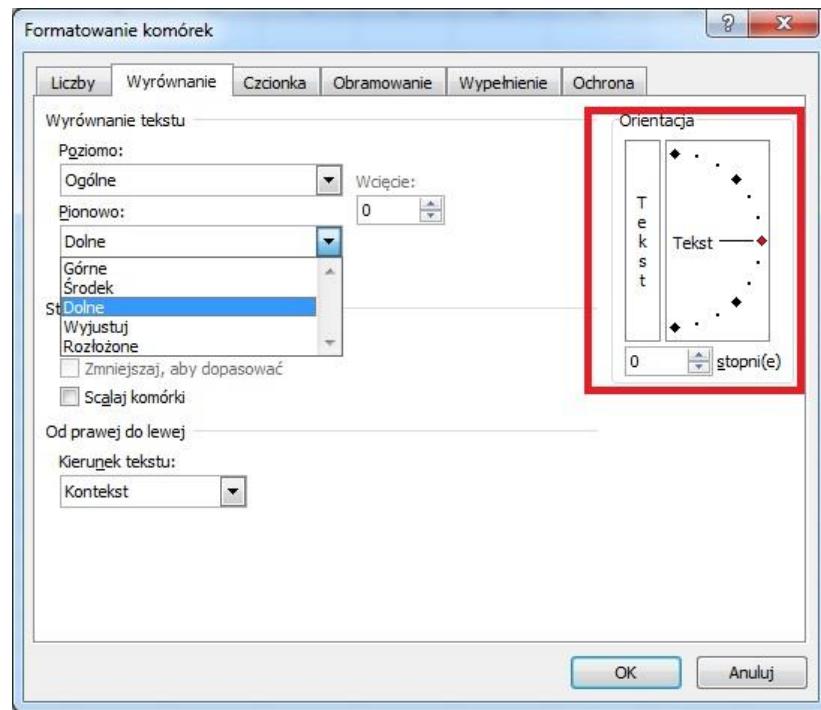


Tekst wpisany do komórek można **wyrównywać na różne sposoby**. Można go wyrównać **w pionie i w poziomie** na kilka sposobów.

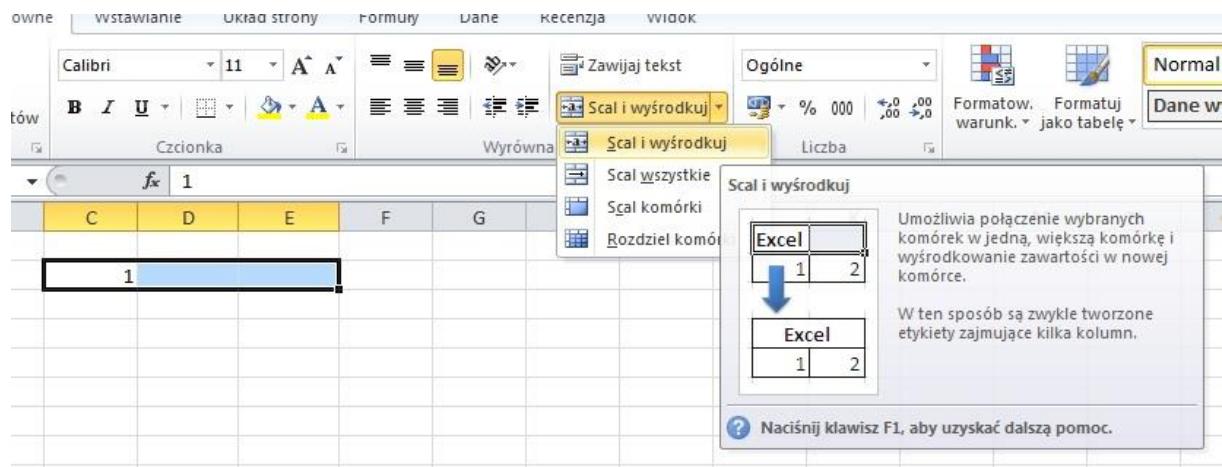




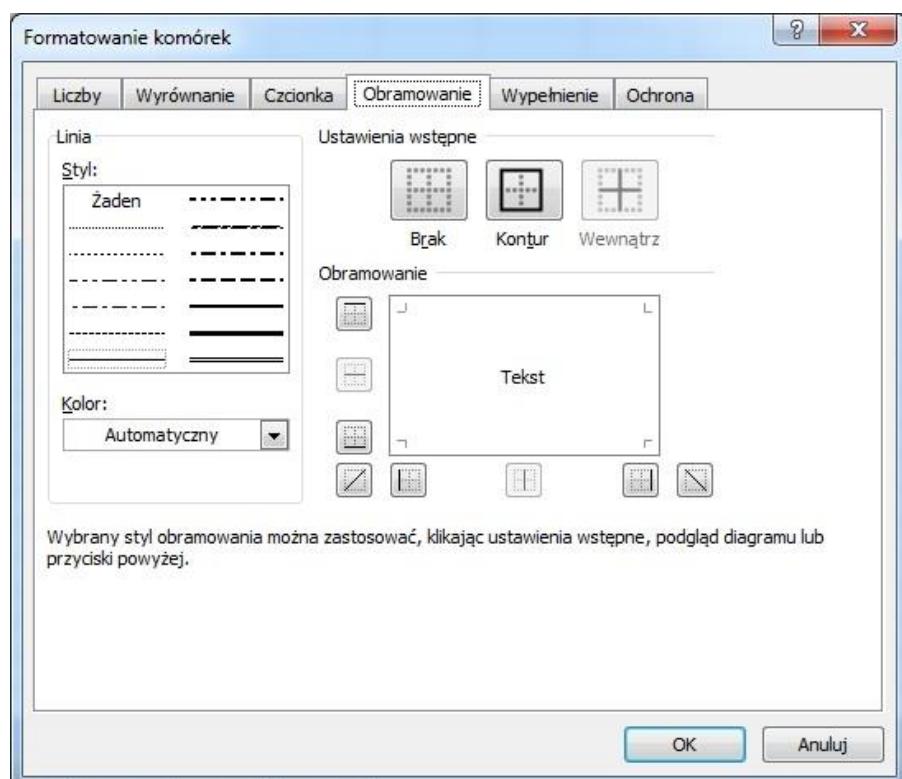
Można również wpisać tekst do komórki pod określonym kątem.

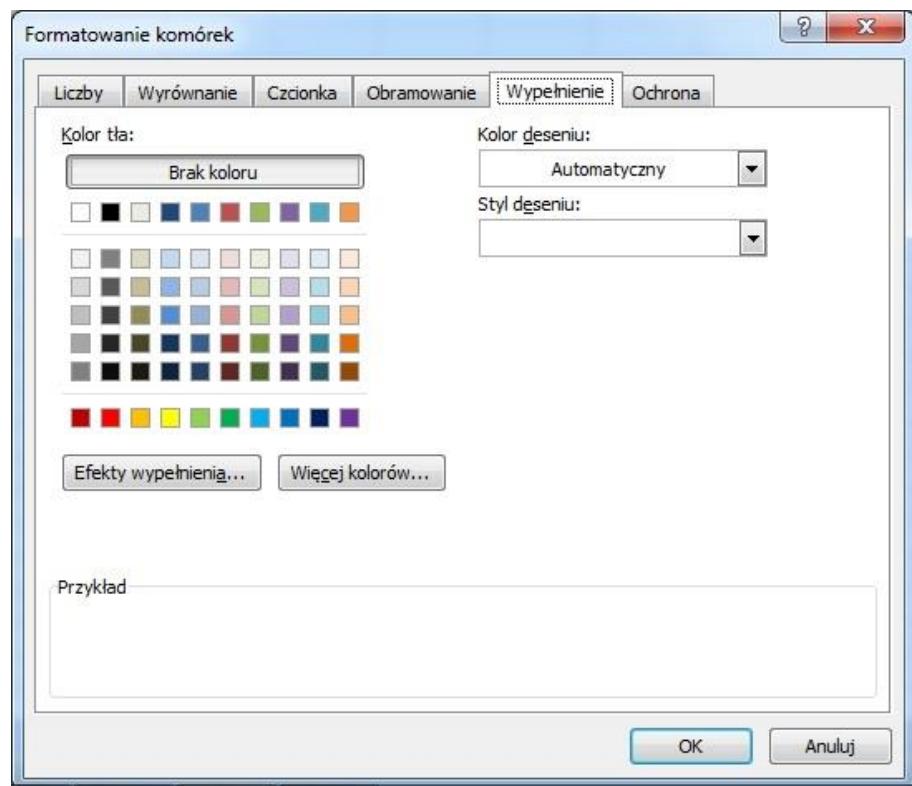


Łączenie ze sobą kilku komórek w jedną nazywa się scalaniem. Można to zrobić wybierając ze wstążki **Wyrównanie->Scal i wyśrodkuj**.



Aby zastosować obramowanie komórki i wypełnienie należy z menu kontekstowego wybrać **Formatuj komórki** następnie skorzystać z zakładek **Obramowanie** i **Wypełnienie**.





Wykresy

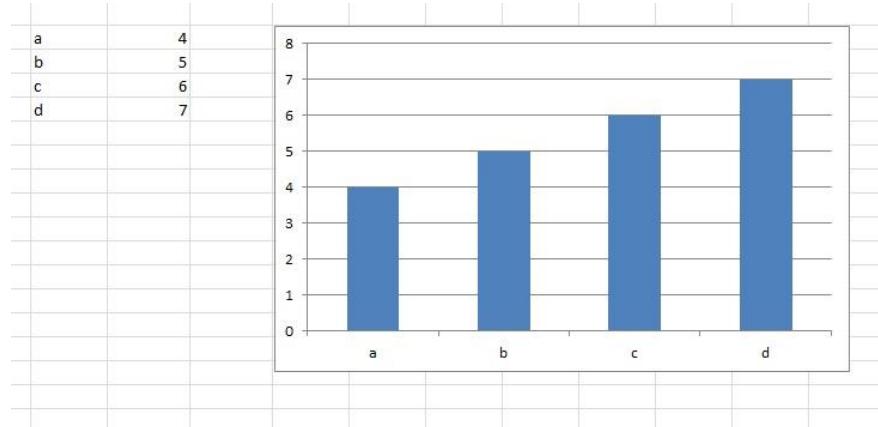
Aby **utworzyć wykres** należy najpierw zaznaczyć komórki, których wykres ma dotyczyć. Następnie wybierzemy typ wykresu. Zrobimy to przy użyciu zakładki **Wykresy** z paska **Wstawianie**.

	C	D	E
a		4	
b		5	
c		6	
d		7	

Wykresy mogą być różnych typów. Do najbardziej **standardowych** wykresów zaliczymy:

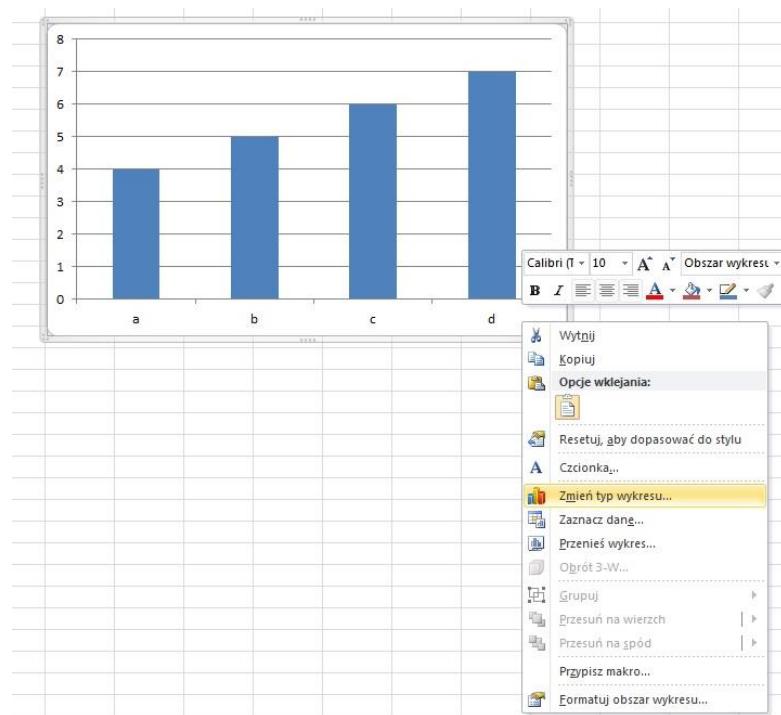
- **kolumnowy**
- **liniowy**
- **kołowy**
- **słupkowy**

Po wybraniu typu wykresu jako kolumnowy otrzymamy wykres podobny do tego jaki jest przedstawiony na poniższym obrazku.



Aby **zaznaczyć wykres** należy go kliknąć lewym klawiszem myszki.

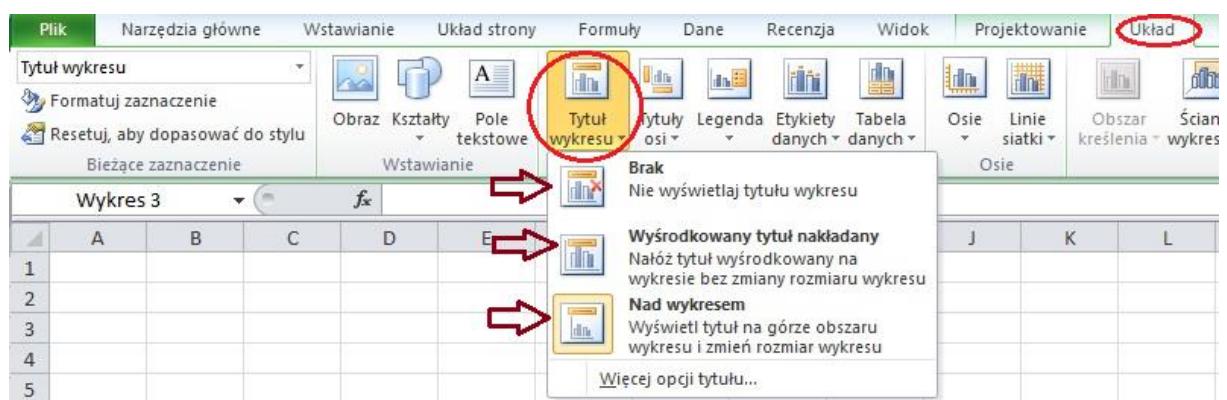
Aby zmienić **rodzaj wykresu** należy go zaznaczyć, następnie z menu kontekstowego wybrać **Zmień typ wykresu**.



Aby przenieś wykres do innego arkusza, należy najpierw go zaznaczyć. Następnie z menu kontekstowego wybierzemy **Przenieś wykres**.

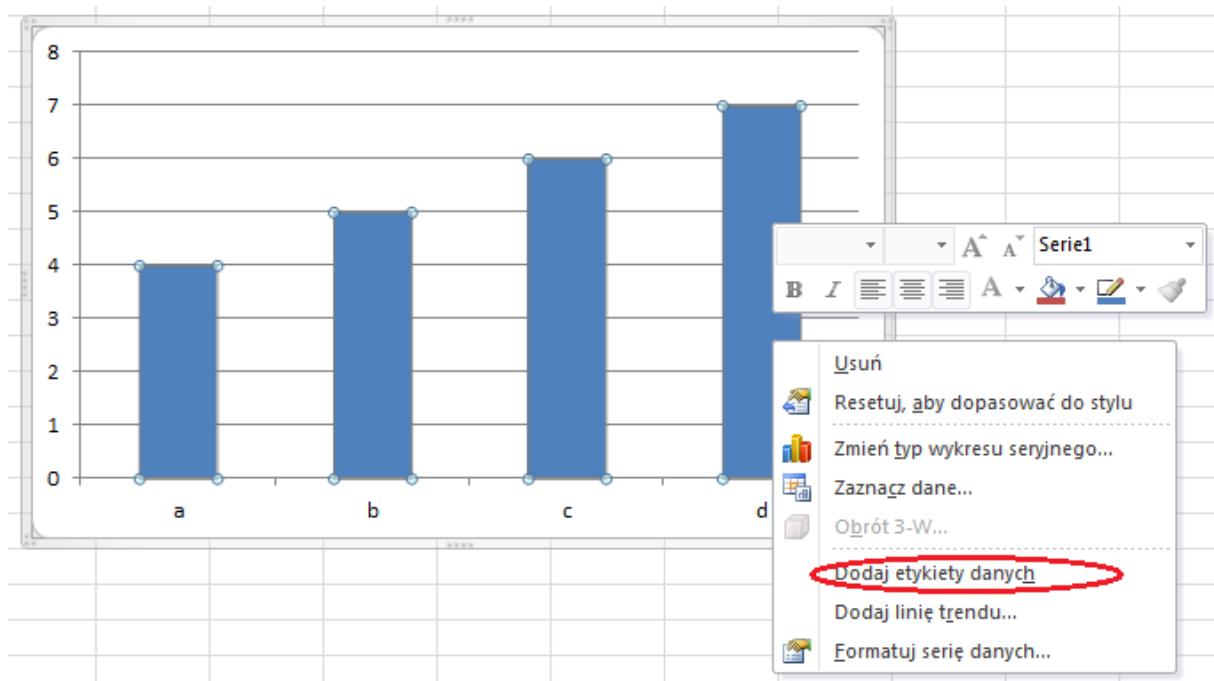
Aby usunąć wykres należy go zaznaczyć, następnie wybrać z menu kontekstowego **Usuń** lub nacisnąć **DELETE**.

Aby **dodać** lub **usunąć pole tytułu wykresu** należy zaznaczyć wykres, następnie z menu głównego wybrać **Układ -> Tytuł wykresu**, a następnie wybrać jedną z wybranych opcji lub otworzyć inne opcje tytułu wykresu wybierając „**więcej opcji tytułu**”.

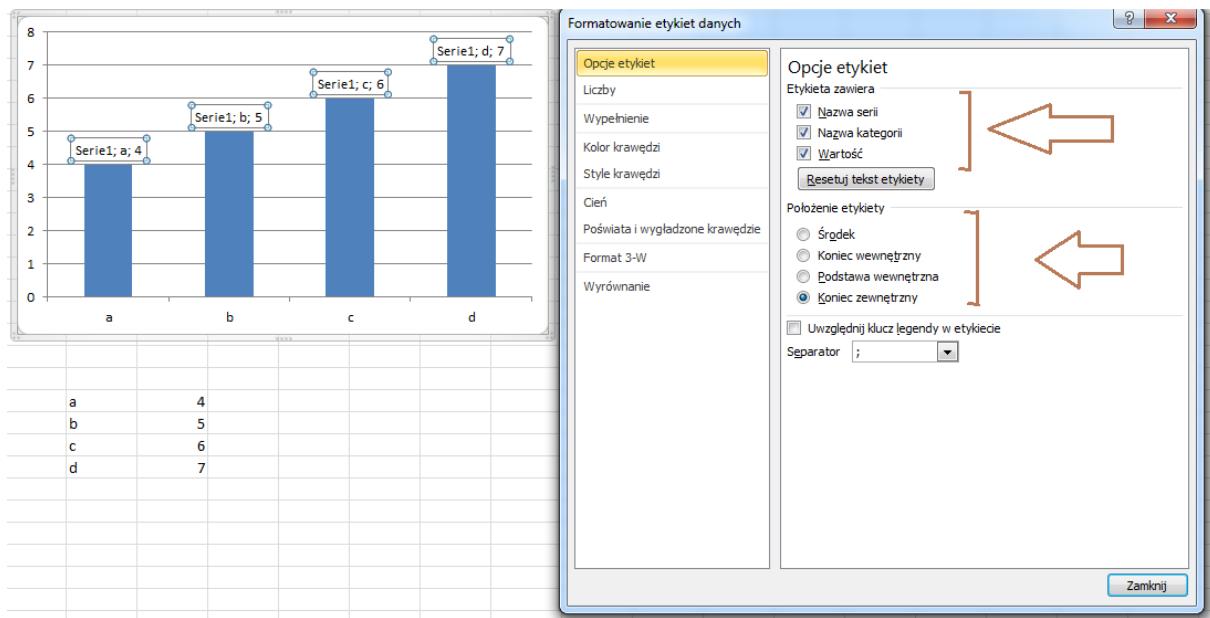


Aby edytować tytuł wykresu należy kliknąć lewym przyciskiem na tytuł wykresu, następnie jeszcze raz kliknąć lewym klawiszem (aby pojawił się wskaźnik tekstu) lub prawym przyciskiem uruchomić menu kontekstowe z którego wybieramy „**edytuj tekst**” i wtedy wpisać odpowiedni tytuł.

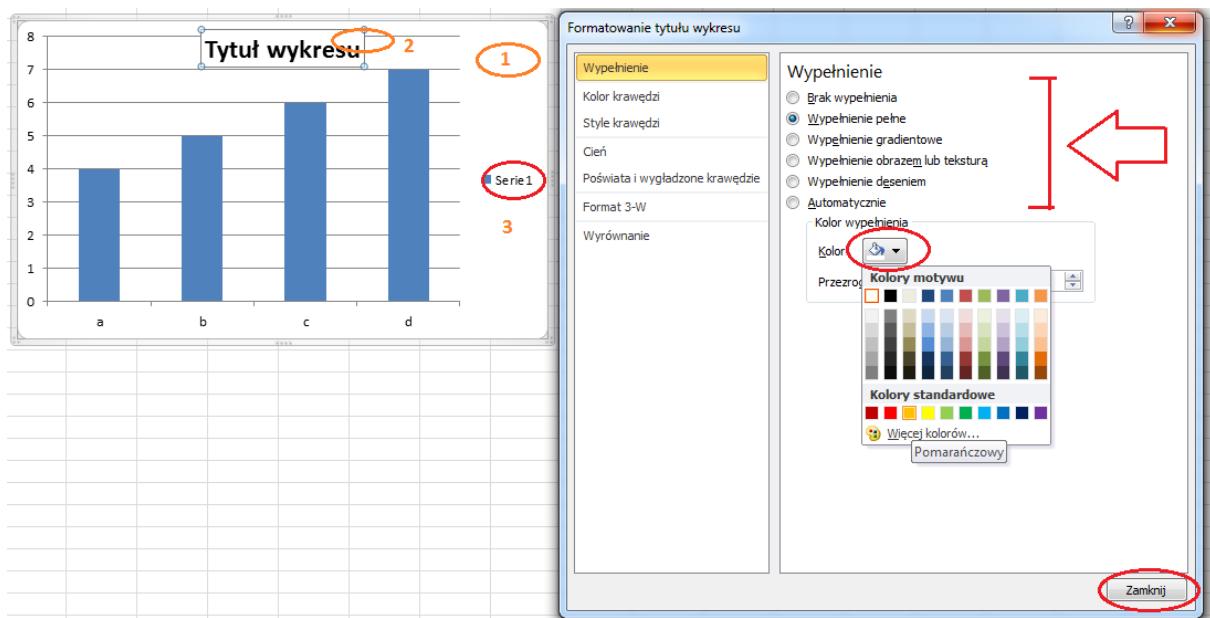
Aby dodać etykiety z danymi do wykresu należy wybrać jedną serię danych. Trzeba wtedy wybrać z menu kontekstowego „**dodaj etykiety danych**”.



Utworzone etykiety można edytować wybierając z menu kontekstowego „**Formatuj etykiety danych**”. W „**Opcji etykiet**” wybieramy elementy jakie mają być wyświetlane, natomiast w opcji „**Liczby**” zaznaczamy w jakiej kategorii mają być pokazywane wartości (Ogólne, Walutowe, Procentowe, Data, ...)

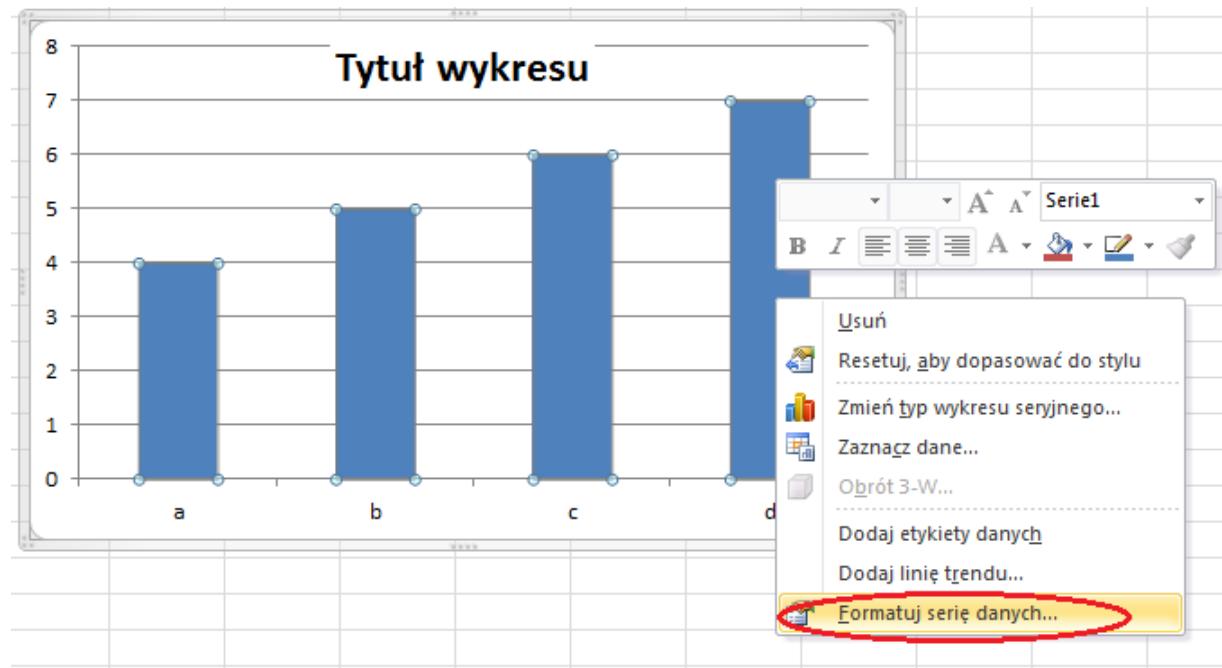


Aby zmienić kolor tła dla wykresu i legendy należy lewym przyciskiem myszy wybrać odpowiedni element wykresu (1 - obszar wykresu, 2 – obszar kreślenia, 3 – legenda) i wybrać z menu kontekstowego „formatuj” (1 – „formatuj obszar wykresu”, 2 – „formatuj obszar kreślenia”, 3 – „formatuj legendę” odpowiednio). W zakładce **Wypełnienie**, zaznaczamy odpowiedni rodzaj wypełnienia lub go wyłączamy (obszar staje się przezroczysty). Następnie wybieramy rodzaj koloru.

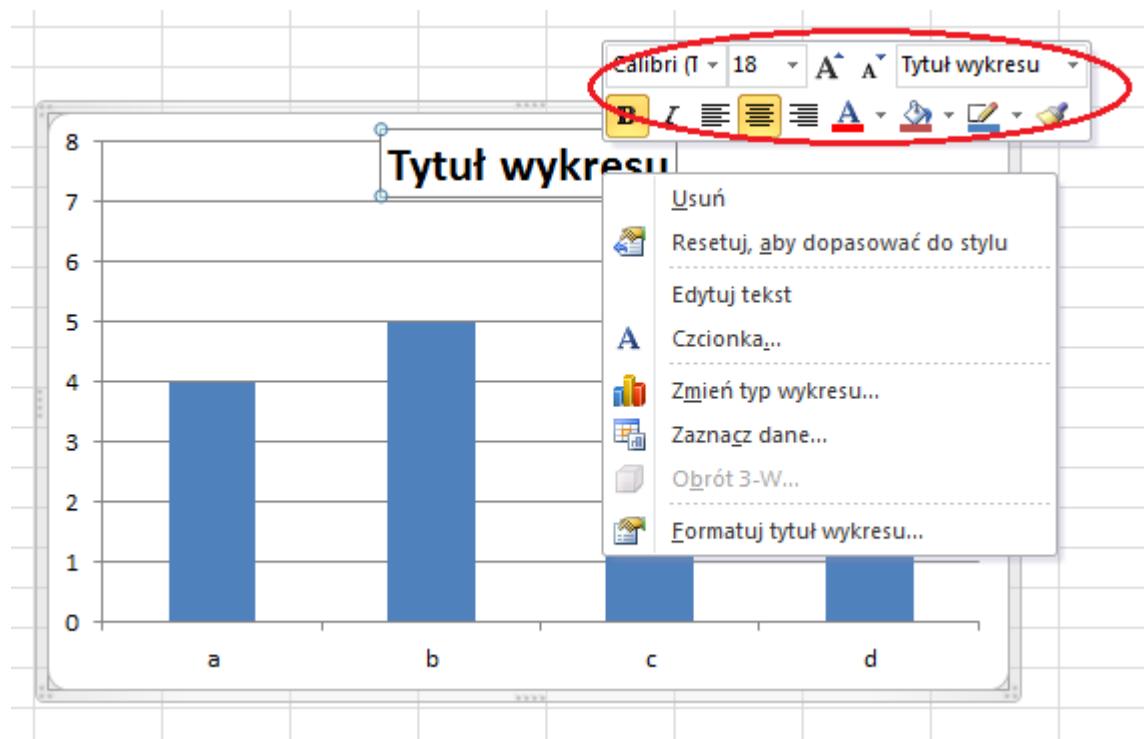


Zaznaczamy wszystkie elementy wybranej serii klikając lewym przyciskiem myszy i wybieramy z menu kontekstowego „Formatuj serię danych ...”. W zakładce „wypełnienie” zaznaczamy odpowiedni rodzaj wypełnienia lub go wyłączamy (obszar staje się przezroczysty).

Następnie wybieramy odpowiedni rodzaj koloru korzystając z zakładek **kolor krawędzi** i **style krawędzi**.

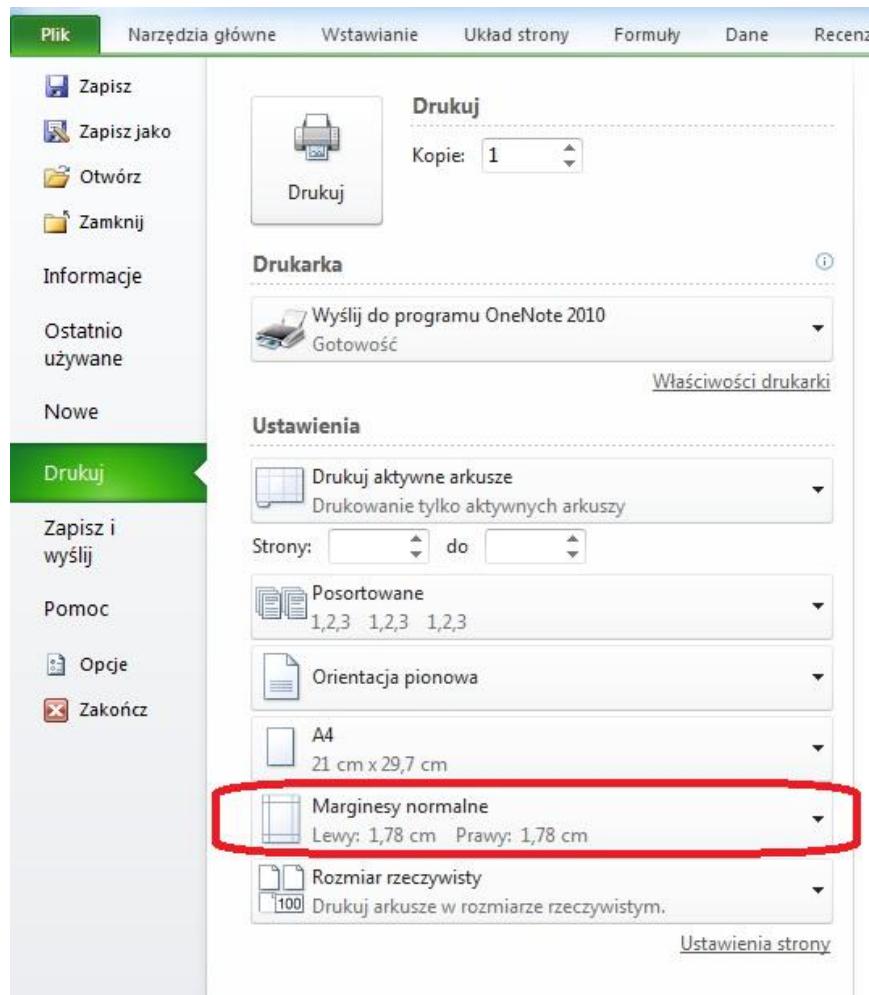


Aby zmienić rozmiar i kolor czcionki dla wybranych elementów wykresu klikamy lewym przyciskiem myszy na wybrany element, następnie używając prawego klawisza myszy otwieramy menu kontekstowe. Zauważmy od razu możliwość zmiany rozmiaru i koloru czcionki.



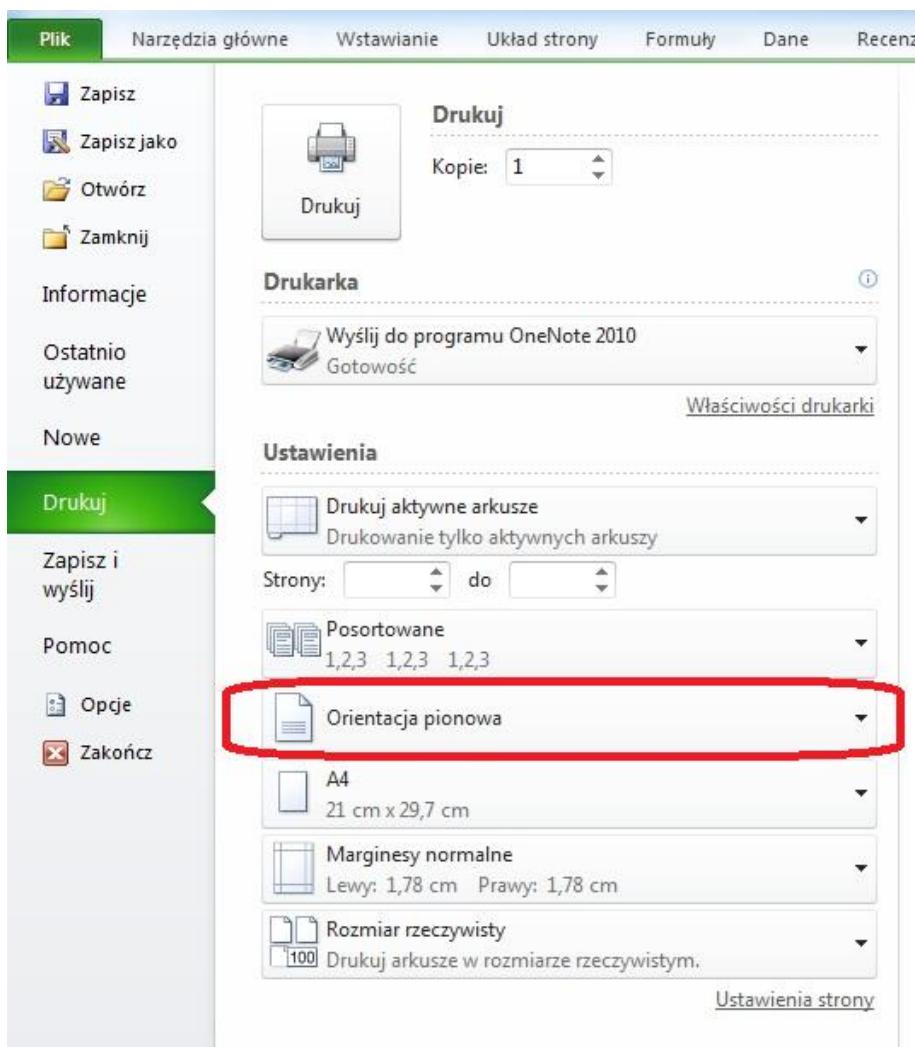
Formatowanie arkusza

Aby zmienić marginesy strony należy z paska narzędzi wybrać **Plik->Drukuj**.



Następnie wybieramy **Marginesy** a później jeden z przykładowych układów marginesów lub wybierając **Marginesy niestandardowe** można je ustawić samodzielnie według własnych preferencji.

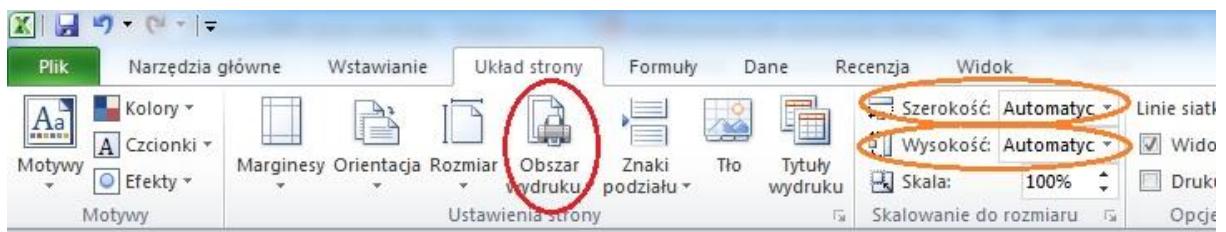
Aby zmienić **orientację dokumentu** należy z paska narzędzi **Plik** wybrać **Drukuj** a następnie wybrać **Orientację**.



Aby zmienić rozmiar papieru należy użyć opcji znajdującej się poniżej Orientacji.

Jak rozmieścić zawartość arkusza na określonej liczbie stron

Zaznaczamy fragment arkusza przeznaczony do wydruku. Z menu „**Układ strony**” wybieramy „**obszar wydruku**” i klikamy na „**ustaw obszar wydruku**”. A następnie uruchamiamy rozmiar skalowania w „skalowanie rozmiaru” wybierając szerokość (ile stron w szerokości) oraz wysokość (ile stron w szerokości).



W menu „**Układ strony**” wybieramy przycisk „**Tytuły wydruku**”. Następnie korzystamy z zakładki **Nagłówek/stopka** aby ustawić nagłówek lub stopkę. Najlepiej jest teraz wybrać jedną z dwóch opcji: „**Nagłówek niestandardowy**” lub „**Stopka niestandardowa**”.

Ustawienia strony

- Strona
- Marginesy
- Nagłówek/stopka**
- Arkusz

Strona 1

Nagłówek:

Strona 1

Nagłówek niestandardowy... (circled in blue)

Stopka niestandardowa...

Stopka:

(brak)

Inne na stronach parzystych i nieparzystych

Inne na pierwszej stronie

Skaluj z dokumentem

Wyrównaj do marginów strony

Drukuj... Podgląd wydruku Opcje...

Nagłówek

Nagłówek

Aby formatować tekst: zaznacz tekst i wybierz przycisk Formatuj tekst.

Aby wstawić numer strony, datę, czas, ścieżkę pliku, nazwę pliku lub nazwę arkusza: umieść punkt wstawiania w polu edycji i wybierz odpowiedni przycisk.

Aby wstawić obraz, naciśnij przycisk Wstaw obraz. Aby formatować obraz, umieść cursor w polu edycji i naciśnij przycisk Formatuj obraz.

Lewa sekcja: Środkowa sekcja: Prawa sekcja:

OK Anuluj

Po wybraniu nagłówka pojawi się nam zakładka **Nagłówek** widoczna na powyższym obrazku.

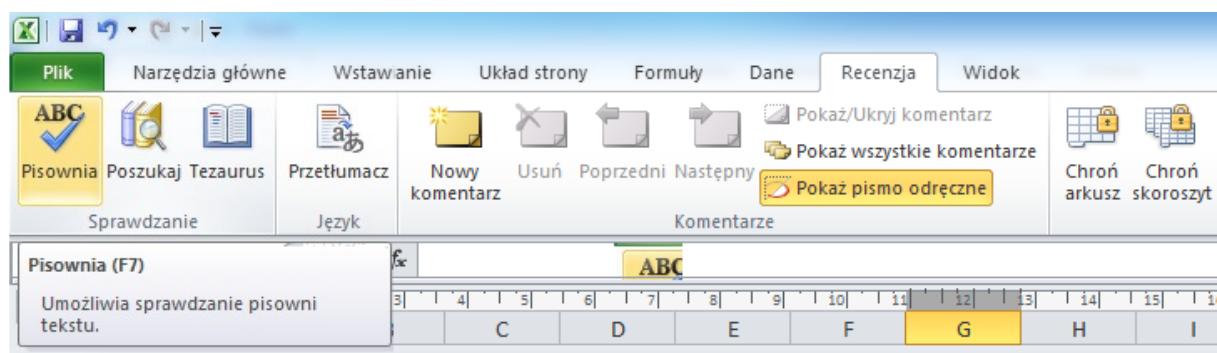
Wybieramy odpowiednią sekcję (kolor niebieski), w której chcemy umieścić tekst. Tekst, który stanie się nagłówkiem można od razu odpowiednio sformatować wybierając znak **A** (kolor zielony).

Aby usunąć stopkę/nagłówek wykonujemy podobne czynności w celu uruchomienia zakładki **Ustawienia strony -> Nagłówek/Stopka** i usuwamy z sekcji (kolor niebieski) wszelkie wpisy.

Aby wstawić i usuwać z nagłówka i stopki elementy należy użyć przycisków otoczonych zielonym kolorem na ostatnim obrazku.

 oznacza numer strony;  ilość stron;  data ;  godzina;  miejsce pliku na dysku;  nazwa pliku;  nazwa arkusza;  rysunek, który możemy edytować wybierając .

Drukowanie, sprawdzenie i poprawa arkusza pod względem rachunkowym i językowym



Zanim postanowimy **wydrukować** nasz arkusz, to musimy **sprawdzić czy nie ma błędów** i je ewentualnie **poprawić**.

Aby **sprawdzić i poprawić błędy pod względem językowym** należy wybrać z paska **Recenzja->Pisownia**.

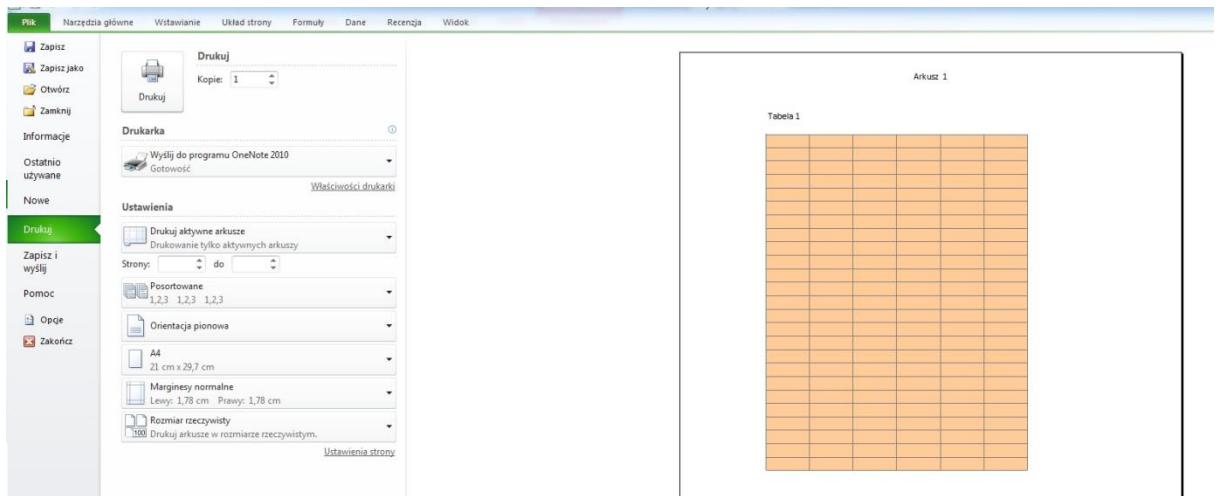
Arkusz będzie sprawdzony **pod względem rachunkowym** jeżeli nie będą w nim występowały żadne błędy takie same lub podobne do błędów opisanych w punkcie 4.1.3.

W celu włączenia drukowania nagłówków wierszy i kolumn należy w pasku „**Układ strony**” wybrać przycisk „**Tytuły wydruku**”. Następnie wybieramy zakładkę „**Arkusz**” i zaznaczamy w „**Drukuj**” wpis „**Nagłówki wierszy i kolumn**”. W podobny sposób włączamy/wyłączamy funkcję drukowania linii siatki („**Drukuj->Linie siatki**”).

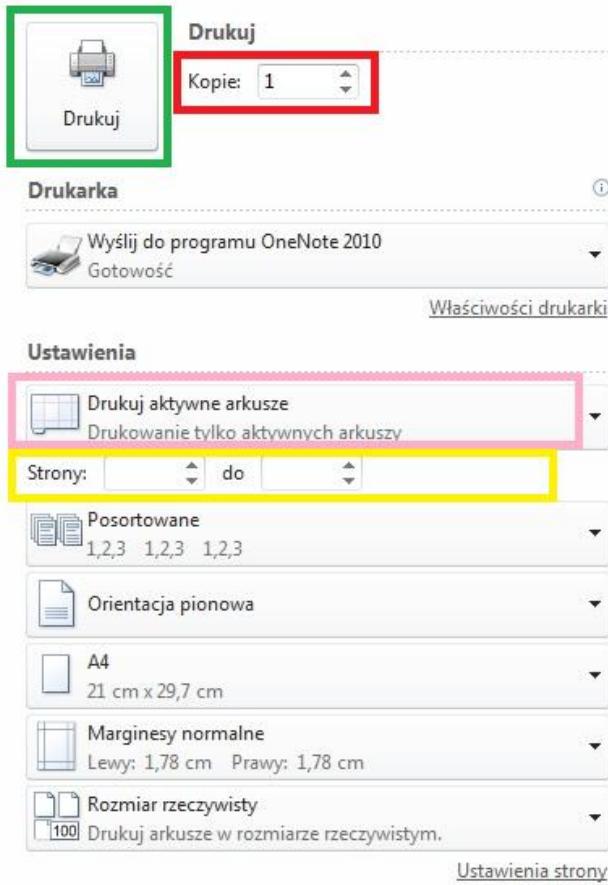
Automatyczne powtarzanie nagłówków kolumn i wierszy na każdej drukowanej stronie arkusza.

Aby zrealizować te czynność należy w zakładce **Arkusz** zaznaczyć „**U góry powtarzaj wiersze**” i/lub dla kolumn „**Z lewej powtarzaj kolumny**”.

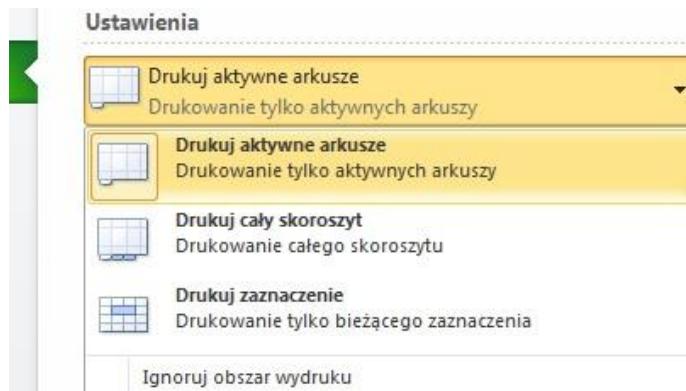
Aby sprawdzić wygląd arkusza przed wydrukiem należy z paska **Plik** wybrać **Drukuj**. Wtedy zobaczymy okienko podobne do poniższego obrazka.



Z lewej strony mamy ustawienia wydruku, a z prawej widzimy **podgląd wydruku arkusza**.



Aby wydrukować cały arkusz należy wybrać **drukuj cały arkusz** (kolor różowy), a następnie nacisnąć przycisk **Drukuj**. Możemy również wydrukować jeden arkusz, lub **wybrany fragment** arkusza.



Jeśli chcemy wydrukować kilka kopii należy wprowadzić odpowiednią liczbę w polu **Kopie** (kolor czerwony). Jeżeli zależy nam tylko na kilku stronach, to w polu **Strony** należy podać ich numery (kolor żółty).

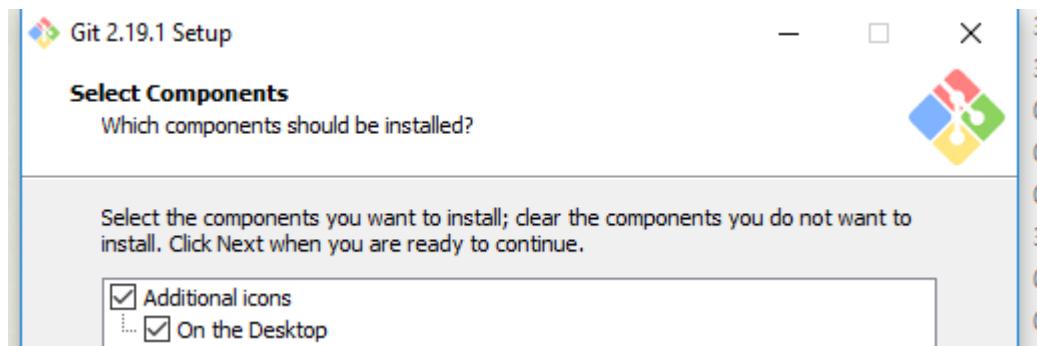
System kontroli wersji – git

Poradnik do gita

1) Założyć konto na github.com

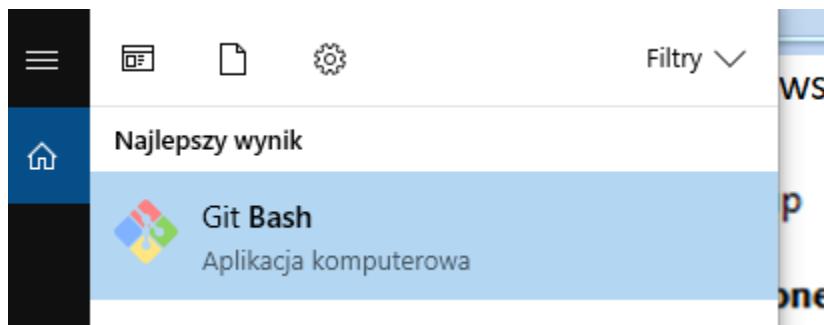
2) Zainstalować <https://git-scm.com/download/win>

Podczas instalacji wszystkie opcje pozostawić takie jakie są poza dwoma kwadracikami



Zaznaczyć te dwie rzeczy

3) Gita można uruchomić klikając w skrót



Albo zaznaczyć katalog i wybrać GithBash Here



4) Po uruchomieniu gita przechodzimy do katalogu z naszym projektem

```
MINGW64:/c/projekty/DrivingLicense
User@SALA205 MINGW64 /c
$ cd c:/projekty
User@SALA205 MINGW64 /c/projekty
$ cd DrivingLicense/
User@SALA205 MINGW64 /c/projekty/DrivingLicense
$
```

5) Praca z gitem

a) inicjalizujemy repozytorium polecienniem **git init**

```
User@SALA205 MINGW64 /c/projekty/DrivingLicense
$ git init
Initialized empty Git repository in c:/projekty/DrivingLicense/.git/
User@SALA205 MINGW64 /c/projekty/DrivingLicense (master)
```

Zwróćmy uwagę na napis **master**. Oznacza, że jesteśmy na tzw. gałęzi głównej. Powinien być obecny również ukryty katalog **.git**. Można to sprawdzić polecienniem **ls -la**.

b) Konfigurujemy gita zgodnie z danymi naszego konta

```
User@SALA205 MINGW64 /c/projekty/DrivingLicense (master)
$ git config --global user.name "Tomasz Idzikowski"

User@SALA205 MINGW64 /c/projekty/DrivingLicense (master)
$ git config --global user.email "idzikdev@gmail.com"
```

Możemy wydać poleciennie **git config -l**, aby sprawdzić czy konfiguracja została zapisana.

c) Zakładamy jakiś plik w tym folderze wydając poleciennie **touch** i sprawdzamy czy plik został założony

```
User@SALA205 MINGW64 /c/projekty/DrivingLicense (master)
$ touch plik

User@SALA205 MINGW64 /c/projekty/DrivingLicense (master)
$ ls -la
total 8
drwxr-xr-x 1 User 197121 0 lis  9 08:44 .
drwxr-xr-x 1 User 197121 0 lis  9 08:37 ..
drwxr-xr-x 1 User 197121 0 lis  9 08:39 .git/
-rw-r--r-- 1 User 197121 0 lis  9 08:44 plik
```

d) Przekażemy ten plik do zdalnego repozytorium

6) Zakładamy repozytorium na naszym koncie na stronie github.com/nickname

Klikamy na Repositories, a następnie na New

The screenshot shows the GitHub user interface for creating a new repository. At the top, there are tabs for 'Overview', 'Repositories 3', 'Stars 0', 'Followers 1', and 'Following 5'. Below these are search filters for 'Type: All' and 'Language: All', and a prominent green 'New' button. A search bar with the placeholder 'Find a repository...' is also visible.

Wypełniamy pole nazwa i ewentualnie opis. Nie wolno zaznaczać Initialize with readme.

The screenshot shows the 'Create new repository' form. It includes fields for 'Owner' (set to 'idzikdev'), 'Repository name' (empty), and a note below stating 'Great repository names are short and memorable. Need inspiration? How about sturdy-fortnight.' There is also a 'Description (optional)' field with an empty text area.

Dostaniemy coś podobnego

The screenshot shows the 'Quick setup — if you've done this kind of thing before' section. It provides instructions for creating a new repository on the command line:

```
echo "# aaa" >> README.md
git init
git add README.md
git commit -m "first commit"
git remote add origin https://github.com/idzikdev/aaa.git
git push -u origin master
```

...or push an existing repository from the command line

```
git remote add origin https://github.com/idzikdev/aaa.git
git push -u origin master
```

Widzimy tutaj instrukcję co po kolei mamy robić

Wróćmy do git'a, aby przesyłać nasz plik do zdalnego repo

5) ciąg dalszy (poleceni, które będą tu podawane widać na powyższym screenie)

e) Pisząc **git status** widzimy, że plik touch nie jest śledzony

```
User@SALA205 MINGW64 /c/projekty/DrivingLicense (master)
$ git status
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be com-
    plik

nothing added to commit but untracked files present (use
```

Dodamy go do śledzenia zmian poleceniem **git add** lub **git add .** (w poleceniu jest kropka) czyli wszystkie)

```
User@SALA205 MINGW64 /c/projekty/DrivingLicense (master)
$ git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)

    new file:   plik
```

f) jak widać zmiany trzeba zacommitować czyli **gitcommit -m „komentarz”**. Pamiętajcie na przyszłość, że jak już kończycie pracę z jakimś plikiem i chcecie żeby inni go zobaczyli, to należy go zacommitować i wysłać na zdalne repo. Można iść dopiero wtedy gdy pisząc git status wszysktko jest OK. ;)

```
User@SALA205 MINGW64 /c/projekty/DrivingLicense (master)
$ git commit -m "initial commmit"
[master (root-commit) 4f6ac84] initial commmit
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 plik

User@SALA205 MINGW64 /c/projekty/DrivingLicense (master)
$ git status
On branch master
nothing to commit, working tree clean
```

g) takie zmiany można cofnąć poleceniem **git reset**, ale założymy, że nie ma błędów i chcemy tego commita wysłać do zdalnego repozytorium. Musimy najpierw określić gdzie jest nasze zdalne repozytorium.

```
User@SALA205 MINGW64 /c/projekty/DrivingLicense (master)
$ git remote add origin https://github.com/idzikdev/aaa.git

User@SALA205 MINGW64 /c/projekty/DrivingLicense (master)
$ git remote -v
origin  https://github.com/idzikdev/aaa.git (fetch)
origin  https://github.com/idzikdev/aaa.git (push)
```

h) pozostało nam te zmiany już tylko wypchnąć do repo. Używamy do tego polecenie **push**.

Jeśli pierwsz razy wysyłamy commita do tego repo, to musimy użyć polecenia push z parametrem czyli **push -u**. Następnym razem wystarczy samo **git push**. Jeśli pierwszy raz wysyłamy pliki do jakiegoś naszego repo, to musimy się zautoryzować. Na pewno na pasku zadań wyskoczy jakieś okienko do wprowadzania hasła.

```
User@SALA205 MINGW64 /c/projekty/DrivingLicense (master)
$ git push -u origin master
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Writing objects: 100% (3/3), 216 bytes | 216.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0)
remote:
remote: Create a pull request for 'master' on GitHub by visitir
remote:      https://github.com/idzikdev/aaa/pull/new/master
remote:
To https://github.com/idzikdev/aaa.git
 * [new branch]      master -> master
Branch 'master' set up to track remote branch 'master' from 'or

User@SALA205 MINGW64 /c/projekty/DrivingLicense (master)
```

I to tyle!

Na naszym repo w sieci zobaczymy nasz plik

No description, website, or topics provided.

Manage topics

1 commit 1 branch

Branch: master ▾ New pull request

idzikdev initial commmit

plik initial commmit

i) gdybyśmy chcieliściagnąć ze zdalnego repo aktualną wersję plików (a robi się TO ZASZWSZE jak się pracuje na wspólnym repo), to należy wydać polecenie

git pull lub **git fetch**

a później zmergować. Jednak dla początkujących polecam **git pull**.

```
User@SALA205 MINGW64 /c/projekty/DrivingLicense (master)
$ git pull
Already up to date.
```

j) Jeśli pracujemy na obcym komputerze to najlepiej będzie usunąć swoje hasło do konta gitowskiego czyli tzwcredentials. Robimy to w oknie Konta Użytkownika->Poświadczenie systemu Windows

The screenshot shows a software interface for managing certificates. At the top, there are two tabs: "Po誓iadczenia sieci Web" (selected) and "Po誓iadczenia systemu Windows". Below the tabs, there are three main sections:

- Po誓iadczenia systemu Windows**:
Text: "Brak po誓iadczeń systemu Windows."
Action: "Dodaj po誓iadczenie systemu Windows" (with a blue link)
- Po誓iadczenia oparte na certyfikatach**:
Text: "Brak certyfikatów."
Action: "Dodaj po誓iadczenie oparte na certyfikacie" (with a blue link)
- Po誓iadczenia rodzajowe**:
Text: "git:https://github.com"
Text: "Zmodyfikowano: Dzisiaj" (with a blue link)

Dodatki

- 1) Pierwsze pobieranie ze zdalnego repo - **git checkout --trackorigin/master**
- 2) Kasowanie credentialsname - **git config --global --unset-all user.name**
- 3) Kasowanie credentials email - **git config --global --unset-alluser.email**
- 4) Zapamiętywanie credentials - **git config credential.helperstore**