Pseudo kod Algorytmu

## Spis Treści

1. Pełne Definiowanie Zmiennych Globalnych

## // PEŁNE DEFINIOWANIE ZMIENNYCH GLOBALNYCH

DEFINIUJ ZMIENNE GLOBALNE:

stan\_symulacji // Stan postępu symulacji (0% do 100%)

licznik\_symulacji // Licznik symulacji

liczba\_symulacji // Liczba wykonywanych symulacji

szkody\_zapis\_step // Krok zapisu szkód w procentach

save\_step // Krok zapisu danych do plików wynikowych

exponsure\_latitude // Szerokość geograficzna budynków

exponsure\_longitude // Długość geograficzna budynków

exponsure\_insurance // Ubezpieczenia budynków

exponsure\_reassurance // Reasekuracje budynków

exponsure\_sum\_value // Wartości ubezpieczonych budynków

list\_list\_wyb // Prawdopodobieństwa powstania pożaru

fire\_spread\_prob\_vec // Prawdopodobieństwa rozprzestrzeniania się pożaru

wielkosc\_pozaru // Rozmiary pożarów i ich prawdopodobieństwa

mutex\_pierwotny // Mutex dla bufora pożarów pierwotnych

mutex\_rozprz // Mutex dla bufora pożarów rozprzestrzenionych

cv\_pierwotny // Zmienna warunkowa dla pożarów pierwotnych

cv\_rozprz // Zmienna warunkowa dla pożarów rozprzestrzenionych

2. Pełne Definiowanie Struktur Danych

## // PEŁNE DEFINIOWANIE STRUKTUR DANYCH

STRUKTURA Data:

- longitude // Długość geograficzna budynku

- latitude // Szerokość geograficzna budynku

- wartosc // Wartość budynku

- ubezpieczenie // Kwota ubezpieczenia

- reasekuracja // Kwota reasekuracji

KONIEC STRUKTURY

3. Pełne Definiowanie Klas i Relacji

## // PEŁNE DEFINIOWANIE KLAS I RELACJI

INTERFEJS IPozar:

- oblicz\_szkody // Oblicza szkody dla pożaru

- dodaj\_do\_bufora // Dodaje pożar do bufora

- ustaw\_ubezpieczyciela // Przypisuje pożar do ubezpieczyciela

KONIEC INTERFEJSU

KLASA PozarPierwotny IMPLEMENTUJE IPozar:

- budynek // Indeks budynku, który zapłonął

- wielkosc // Wielkość pożaru (%)

- szkoda // Kwota szkody

KONIEC KLASY

4. Wczytywanie danych

### // Funkcja: Wczytywanie danych geograficznych z pliku CSV FUNKCJA wczytajDaneGeograficzne(filePath) OTWÓRZ PLIK filePath JAKO csvin // Mapa przechowująca dane z wiersza CSV ZMIENNA: row <- MAPA <KLUCZ, WARTOŚĆ> // Wczytywanie danych z pliku CSV DOPÓKI NIE KONIEC PLIKU Wczytaj WIERSZ jako row ZMIENNA: region <- KONWERSJA NA INT(row["Region"]) ZMIENNA: latitude <- KONWERSJA NA DOUBLE(row["Latitude"]) ZMIENNA: longitude <- KONWERSJA NA DOUBLE(row["Longitude"]) ZMIENNA: insurance <- KONWERSJA NA INT(row["Insurance"]) ZMIENNA: reassurance <- KONWERSJA NA INT(row["Reassurance"]) ZMIENNA: sumValue <- KONWERSJA NA DOUBLE(row["SumValue"]) ZMIENNA: startDate <- row["StartDate"] ZMIENNA: endDate <- row["EndDate"] // Wyodrębnienie miesięcy z dat ZMIENNA: startMonth <- extractMonth(startDate) - 1 ZMIENNA: endMonth <- extractMonth(endDate) - 1 // Przypisanie danych do odpowiednich wektorów DLA KAŻDEGO miesiąca OD startMonth DO endMonth exponsure\_latitude[region][miesiąc] <- latitude exponsure\_longitude[region][miesiąc] <- longitude exponsure\_insurance[region][miesiąc] <- insurance exponsure\_reassurance[region][miesiąc] <- reassurance exponsure\_sum\_value[region][miesiąc] <- sumValue KONIEC DLA KONIEC DOPÓKI ZAMKNIJ PLIK KONIEC FUNKCJI

5.1. Wczytywanie danych o prawdopodobieństwach

### // Funkcja: Wczytywanie danych o prawdopodobieństwach rozprzestrzeniania się pożaru FUNKCJA wczytajPrRozprzestrzenienia(filePath) OTWÓRZ PLIK filePath JAKO csvin // Mapa przechowująca dane z wiersza CSV ZMIENNA: row <- MAPA <KLUCZ, WARTOŚĆ> ZMIENNA: cnt <- 0 DOPÓKI NIE KONIEC PLIKU Wczytaj WIERSZ jako row fire\_spread\_prob\_vec[cnt] <- PUSTA LISTA // Dodawanie wartości do listy prawdopodobieństw DODAJ DO fire\_spread\_prob\_vec[cnt] <- KONWERSJA NA DOUBLE(row["0"]) DODAJ DO fire\_spread\_prob\_vec[cnt] <- KONWERSJA NA DOUBLE(row["(0,25]"]) DODAJ DO fire\_spread\_prob\_vec[cnt] <- KONWERSJA NA DOUBLE(row["(25,50]"]) DODAJ DO fire\_spread\_prob\_vec[cnt] <- KONWERSJA NA DOUBLE(row["(50,75]"]) DODAJ DO fire\_spread\_prob\_vec[cnt] <- KONWERSJA NA DOUBLE(row["(75,100]"]) DODAJ DO fire\_spread\_prob\_vec[cnt] <- KONWERSJA NA DOUBLE(row["(100,125]"]) DODAJ DO fire\_spread\_prob\_vec[cnt] <- KONWERSJA NA DOUBLE(row["(125,150]"]) DODAJ DO fire\_spread\_prob\_vec[cnt] <- KONWERSJA NA DOUBLE(row["(150,175]"]) DODAJ DO fire\_spread\_prob\_vec[cnt] <- KONWERSJA NA DOUBLE(row["(175,200]"]) cnt <- cnt + 1 KONIEC DOPÓKI ZAMKNIJ PLIK KONIEC FUNKCJI

5.2. Wczytywanie danych o wielkości pożaru

### // Funkcja: Wczytywanie danych o wielkości pożaru

### FUNKCJA wczytajPrWielkoscPozaru(filePath)

OTWÓRZ PLIK filePath JAKO csvin

// Mapa przechowująca dane z wiersza CSV

ZMIENNA: row <- MAPA <KLUCZ, WARTOŚĆ>

DOPÓKI NIE KONIEC PLIKU

Wczytaj WIERSZ jako row

// Dodawanie wartości do listy wielkości pożaru

wielkosc\_pozaru[0] <- KONWERSJA NA DOUBLE(row["Rozmiar"])

wielkosc\_pozaru[1] <- KONWERSJA NA DOUBLE(row["Prawdopodobienstwo"])

KONIEC DOPÓKI

ZAMKNIJ PLIK

KONIEC FUNKCJI

5.4. Wczytywanie danych o reasekuracji

### // Funkcja: Wczytywanie danych obligatoryjnych i fakultatywnych

### FUNKCJA wczytajDaneObligatoryjne(FOLDER\_REAS, nazwy\_plikow)

DLA KAŻDEGO i OD 0 DO ROZMIAR(nazwy\_plikow)

DODAJ PUSTĄ LISTĘ DO obligatoryjna\_input\_risk

DODAJ PUSTĄ LISTĘ DO obligatoryjna\_input\_event

// Inicjalizacja wartości 0 dla każdej kategorii

DLA j OD 0 DO 3

obligatoryjna\_input\_risk[i].DODAJ(0)

obligatoryjna\_input\_event[i].DODAJ(0)

KONIEC DLA

KONIEC DLA

// Wczytywanie danych z plików CSV dla każdego ubezpieczyciela

DLA KAŻDEGO i OD 0 DO ROZMIAR(nazwy\_plikow)

OTWÓRZ PLIK FOLDER\_REAS + nazwy\_plikow[i] + ".csv" JAKO csvin

ZMIENNA: row <- MAPA <KLUCZ, WARTOŚĆ>

ZMIENNA: cnt <- 0

DOPÓKI NIE KONIEC PLIKU

Wczytaj WIERSZ jako row

JEŚLI cnt == 0 TO

obligatoryjna\_input\_risk[i][3] <- KONWERSJA NA DOUBLE(row["Udzial(ryzyko)"])

obligatoryjna\_input\_event[i][3] <- KONWERSJA NA DOUBLE(row["Udzial(zdarzenie)"])

INACZEJ

obligatoryjna\_input\_risk[i][2] <- KONWERSJA NA DOUBLE(row["Udzial(ryzyko)"])

obligatoryjna\_input\_event[i][2] <- KONWERSJA NA DOUBLE(row["Udzial(zdarzenie)"])

KONIEC JEŚLI

obligatoryjna\_input\_risk[i][0] <- KONWERSJA NA DOUBLE(row["Od(ryzyko)"])

obligatoryjna\_input\_risk[i][1] <- KONWERSJA NA DOUBLE(row["Do(ryzyko)"])

obligatoryjna\_input\_event[i][0] <- KONWERSJA NA DOUBLE(row["Od(zdarzenie)"])

obligatoryjna\_input\_event[i][1] <- KONWERSJA NA DOUBLE(row["Do(zdarzenie)"])

cnt <- cnt + 1

JEŚLI cnt == 2 TO

PRZERWIJ PĘTLĘ

KONIEC JEŚLI

KONIEC DOPÓKI

ZAMKNIJ PLIK

KONIEC DLA

KONIEC FUNKCJI

5.5. Wczytywanie danych o parametrach rozprzestrzenienia

### // Funkcja: Wczytywanie parametrów trendów i kowariancji

### FUNKCJA wczytajParametryTrendow(filePath)

OTWÓRZ PLIK filePath JAKO csvin

ZMIENNA: row <- MAPA <KLUCZ, WARTOŚĆ>

ZMIENNA: cnt <- 0

DOPÓKI NIE KONIEC PLIKU

Wczytaj WIERSZ jako row

// Wczytywanie parametrów trendów

conditional\_mean\_trend\_parameters[0] <- KONWERSJA NA DOUBLE(row["a1"])

conditional\_mean\_trend\_parameters[1] <- KONWERSJA NA DOUBLE(row["b1"])

conditional\_mean\_trend\_parameters[2] <- KONWERSJA NA DOUBLE(row["c1"])

// Wczytywanie parametrów kowariancji

conditional\_Cov[0] <- KONWERSJA NA DOUBLE(row["a2"])

conditional\_Cov[1] <- KONWERSJA NA DOUBLE(row["b2"])

conditional\_Cov[2] <- KONWERSJA NA DOUBLE(row["c2"])

cnt <- cnt + 1

KONIEC DOPÓKI

ZAMKNIJ PLIK

KONIEC FUNKCJI

5.6. Główna funkcja wczytywania danych

// Uzupełniona funkcja główna do wczytywania wszystkich danych

### FUNKCJA wczytajWszystkieDane(FOLDER\_DANYCH)

// Wczytywanie danych geograficznych

wczytajDaneGeograficzne(FOLDER\_DANYCH + "/geograficzne.csv")

// Wczytywanie prawdopodobieństw rozprzestrzeniania

wczytajPrRozprzestrzenienia(FOLDER\_DANYCH + "/pr\_rozprzestrzenienia.csv")

// Wczytywanie danych o wielkości pożaru

wczytajPrWielkoscPozaru(FOLDER\_DANYCH + "/pr\_wielkosc\_pozaru.csv")

// Wczytywanie danych obligatoryjnych i fakultatywnych

ZMIENNA: nazwy\_plikow <- ["obligatoryjne1", "obligatoryjne2"]

wczytajDaneObligatoryjne(FOLDER\_DANYCH + "/obligatoryjne", nazwy\_plikow)

// Wczytywanie parametrów trendów i kowariancji

wczytajParametryTrendow(FOLDER\_DANYCH + "/parametry\_trendow.csv")

KONIEC FUNKCJI

6. SYMULACJE

Definicja funkcji

### FUNKCJA: ustal\_exposure\_number(woj, mies)

#### OPIS: Sprawdza liczbę budynków w danym województwie i miesiącu.

#### WEJŚCIE:

- woj: Numer województwa (0 - 16)

- mies: Numer miesiąca (0 - 11)

#### WYJŚCIE:

- Liczba budynków w danym województwie i miesiącu

#### DZIAŁANIE:

- Pobiera długość wektora exponsure\_longitude[woj][mies]

- Zwraca rozmiar tego wektora jako liczbę budynków

PSEUDOKOD:

- liczba\_budynkow = rozmiar(exponsure\_longitude[woj][mies])

- ZWRÓĆ liczba\_budynkow

### FUNKCJA: losuj\_liczbę\_pożarów(exposure\_number, prawdopodobienstwo)

#### OPIS: Losuje liczbę pożarów na podstawie rozkładu dwumianowego.

#### WEJŚCIE:

- exposure\_number: Liczba budynków w danym województwie i miesiącu

- prawdopodobienstwo: Prawdopodobieństwo wystąpienia pożaru

#### WYJŚCIE:

- Liczba pożarów (binom\_fire)

#### DZIAŁANIE:

- Użycie rozkładu dwumianowego do wylosowania liczby pożarów

- Uwzględnia prawdopodobieństwo dla danego miesiąca i województwa

PSEUDOKOD:

- binom\_fire = rozkład\_dwumianowy(exposure\_number, prawdopodobienstwo)

- ZWRÓĆ binom\_fire

### FUNKCJA: wybierz\_zródła\_pożaru(binom\_fire, exposure\_number)

#### OPIS: Losowo wybiera budynki jako źródła pożaru.

#### WEJŚCIE:

- binom\_fire: Liczba pożarów do wygenerowania

- exposure\_number: Liczba budynków w danym województwie i miesiącu

#### WYJŚCIE:

- fire\_sources\_list: Lista indeksów budynków, które są źródłami pożaru

#### DZIAŁANIE:

- Tworzy listę indeksów budynków (od 0 do exposure\_number - 1)

- Losowo wybiera binom\_fire elementów z tej listy

PSEUDOKOD:

- pom\_index\_fire = [0, 1, ..., exposure\_number - 1]

- fire\_sources\_list = losuj\_elementy(pom\_index\_fire, binom\_fire)

- ZWRÓĆ fire\_sources\_list

### FUNKCJA: procent\_szkody(wielkosc\_pozaru)

#### OPIS: Oblicza procent szkody na podstawie wielkości pożaru.

#### WEJŚCIE:

- wielkosc\_pozaru: Rozmiar pożaru na podstawie rozkładu prawdopodobieństwa

#### WYJŚCIE:

- procent\_szkody: Procent szkody spowodowanej pożarem

#### DZIAŁANIE:

- Pobiera losową wartość z rozkładu wielkosc\_pozaru

- Oblicza procent szkody na podstawie tej wartości

PSEUDOKOD:

- procent\_szkody = losowa\_wartosc(wielkosc\_pozaru)

- ZWRÓĆ procent\_szkody

### FUNKCJA: oblicz\_reasekuracje(wielkosc\_pozar\_kwota, woj, mies, nr\_budynku)

#### OPIS: Oblicza kwotę reasekuracji dla danego pożaru.

#### WEJŚCIE:

- wielkosc\_pozar\_kwota: Kwota szkody wywołanej pożarem

- woj: Numer województwa

- mies: Numer miesiąca

- nr\_budynku: Indeks budynku

#### WYJŚCIE:

- reas\_fire: Kwota reasekuracji

#### DZIAŁANIE:

- Pobiera dane o reasekuracji dla budynku

- Oblicza kwotę reasekuracji na podstawie szkody i umów reasekuracyjnych

PSEUDOKOD:

- ubezpieczenie = exponsure\_insurance[woj][mies][nr\_budynku]

- reasekuracja = exponsure\_reassurance[woj][mies][nr\_budynku]

- reas\_fire = MIN(wielkosc\_pozar\_kwota, reasekuracja)

- ZWRÓĆ reas\_fire

### FUNKCJA: oblicz\_rozprzestrzenianie\_pożaru(promien, nr\_budynku, woj, mies)

#### OPIS: Oblicza rozprzestrzenianie się pożaru na sąsiednie budynki.

#### WEJŚCIE:

- promien: Promień oddziaływania pożaru

- nr\_budynku: Indeks budynku będącego źródłem pożaru

- woj: Numer województwa

- mies: Numer miesiąca

#### WYJŚCIE:

- spread\_one\_building: Lista budynków dotkniętych rozprzestrzenianiem się pożaru

#### DZIAŁANIE:

- Użycie algorytmu Haversine do obliczenia odległości między budynkami

- Sprawdzenie czy budynki są w zasięgu promienia oddziaływania pożaru

PSEUDOKOD:

- spread\_one\_building = haversine\_loop\_cpp\_vec(promien, nr\_budynku, woj, mies)

- ZWRÓĆ spread\_one\_building

### FUNKCJA: przeniesDoBuforPierwotny(buildPierwotny, sciezka, nazwa\_pliku)

#### OPIS: Zapisuje dane o pożarze pierwotnym do pliku CSV.

#### WEJŚCIE:

- buildPierwotny: Bufor z danymi o pożarach pierwotnych

- sciezka: Ścieżka katalogu do zapisu

- nazwa\_pliku: Nazwa pliku CSV

#### DZIAŁANIE:

- Przenosi dane z bufora do globalnego bufora zapisu

- Zapisuje dane do pliku CSV w podanej ścieżce

PSEUDOKOD:

- global\_buffer\_pierwotny.dodaj(buildPierwotny)

- ZAPISZ\_DO\_PLIKU(global\_buffer\_pierwotny, sciezka + nazwa\_pliku)

// 1. Inicjalizacja zmiennych i ustawienia początkowe

- Ustal step\_size = 1.0 / (17 \* 12 + ilosc\_ubezpieczycieli)

// Krok dla paska postępu uwzględniający liczbę województw, miesięcy oraz liczbyubezpieczycieli.

- Ustal bar\_step = (1.0 / ((17 + ilosc\_ubezpieczycieli) \* sim))

// Krok dla ogólnego stanu symulacji uwzględniający liczbę symulacji.

- Ustal index\_table = 0

// Licznik zdarzeń pożarowych.

- Inicjalizuj obiekty dla zapisu pożarów:

- buildPierwotny // Bufor dla pożarów pierwotnych.

- buildRozprzestrzeniony // Bufor dla pożarów rozprzestrzenionych.

- Ustal progressbar[num\_watku] = 0.0

// Inicjalizacja paska postępu dla wątku.

// 2. Główna pętla symulacji po województwach i miesiącach

DLA każdego województwa woj od 0 DO 16:

// Pętla główna - przetwarzanie wszystkich 17 województw

- INICJALIZACJA: Reset zmiennych lokalnych dla danego województwa

- exposure\_number = 0

- binom\_fire = 0

- fire\_sources\_list = []

- INFO: Rozpoczęcie przetwarzania województwa woj

- Przykład: "Przetwarzanie województwa: 1"

DLA każdego miesiąca mies od 0 DO 11:

// Pętla wewnętrzna - przetwarzanie wszystkich 12 miesięcy

- INICJALIZACJA: Reset zmiennych lokalnych dla danego miesiąca

- wielkosc\_pozar\_procent = 0.0

- wielkosc\_pozar\_kwota = 0.0

- insurancer = -1

- reas\_fire = 0.0

- INFO: Rozpoczęcie przetwarzania miesiąca mies

- Przykład: "Przetwarzanie miesiąca: 1"

- Ustal exposure\_number = rozmiar(exponsure\_longitude[woj][mies])

// Liczba budynków w danym województwie i miesiącu.

- Jeśli exposure\_number = 0, to nie ma budynków do przetworzenia

JEŻELI exposure\_number > 0 TO:

// Sprawdzenie czy w danym miesiącu są budynki do przetworzenia

- Ustal binom\_fire = losuj\_liczbę\_pożarów(exposure\_number, list\_list\_wyb[woj][mies])

// Losowa liczba pożarów na podstawie rozkładu dwumianowego

- Uwzględnienie prawdopodobieństwa pożaru dla danego miesiąca i województwa

JEŻELI binom\_fire > 0 TO:

// Wystąpił co najmniej jeden pożar w danym miesiącu i województwie

- Ustal fire\_sources\_list = wybierz\_zródła\_pożaru(binom\_fire, exposure\_number)

// Losowy wybór budynków jako źródeł pożaru

- Każdy budynek ma równą szansę na stanie się źródłem pożaru

- INFO: Wybrano budynki jako źródła pożaru

- Przykład: "Budynki źródłowe: [2, 5, 7]"

DLA każdego nr\_budynku w fire\_sources\_list:

// Przetwarzanie każdego budynku, który zapłonął

- index\_table += 1

// Inkrementacja indeksu zdarzeń pożarowych.

- Używane do identyfikacji każdego zdarzenia pożarowego

- INFO: Rozpoczęcie przetwarzania budynku nr\_budynku

- Przykład: "Przetwarzanie budynku: 2"

- Ustal wielkosc\_pozar\_procent = procent\_szkody(wielkosc\_pozaru)

// Obliczenie procentowej wielkości szkody

- Na podstawie rozkładu prawdopodobieństwa wielkości pożaru

- Ustal wielkosc\_pozar\_kwota = wielkosc\_pozar\_procent \* exponsure\_sum\_value[woj][mies][nr\_budynku]

// Obliczenie kwoty szkody na podstawie wartości budynku

- JEŻELI wielkosc\_pozar\_kwota < 500 TO:

// Minimalna wartość szkody wynosi 500

- Ustal wielkosc\_pozar\_kwota = 500

- Ustal insurancer = exponsure\_insurance[woj][mies][nr\_budynku]

// Ustal ubezpieczyciela odpowiedzialnego za budynek

- Ustal reas\_fire = oblicz\_reasekuracje(wielkosc\_pozar\_kwota, woj, mies, nr\_budynku)

// Obliczenie reasekuracji dla danego pożaru

- Uwzględnienie umów reasekuracyjnych ubezpieczyciela

- Dodaj pożar pierwotny do bufora:

- buildPierwotny.addPozarPierwotny(insurancer, nr\_budynku, woj + 1, mies + 1, index\_table, wielkosc\_pozar\_kwota, reas\_fire)

// Buforowanie danych o pożarze pierwotnym

// 2.1. Zapis pożaru pierwotnego

- JEŻELI forma\_zapisu\_budynkow == 0 TO:

- przeniesDoBuforPierwotny(przeniesienie buildPierwotny, nazwakatalogu + "/Pierwotny/", numer\_symulacji + ".csv")

// Natychmiastowe zapisanie pożaru pierwotnego do pliku CSV.

- Ustal spread\_one\_building = oblicz\_rozprzestrzenianie\_pożaru(promien, nr\_budynku, woj, mies)

// Obliczenie rozprzestrzeniania się pożaru

- Użycie algorytmu Haversine do obliczenia wpływu na sąsiednie budynki

- Ustal len\_spread = rozmiar(spread\_one\_building[4])

// Liczba budynków, na które rozprzestrzenił się pożar

JEŻELI len\_spread > 0 TO:

// Pożar rozprzestrzenił się na inne budynki

- Dodaj pożar rozprzestrzeniony do bufora:

- buildRozprzestrzeniony.addPozarRozprzestrzeniony(spread\_one\_building)

// Buforowanie danych o pożarze rozprzestrzenionym

// 2.2. Zapis pożaru rozprzestrzenionego

- JEŻELI forma\_zapisu\_budynkow == 0 TO:

- przeniesDoBuforRozprz(przeniesienie buildRozprzestrzeniony, nazwakatalogu + "/Rozprzestrzeniony/", numer\_symulacji + ".csv")

// Natychmiastowe zapisanie pożaru rozprzestrzenionego do pliku CSV.

- progressbar[num\_watku] += step\_size

// Aktualizacja paska postępu dla bieżącego wątku

- Informuje o postępie przetwarzania dla danego wątku

- stanSymulacji += bar\_step

// Aktualizacja ogólnego stanu symulacji

- Informuje o postępie całej symulacji

## Zapis Szkód w Symulacji

Wątki odpowiedzialne za zapisywanie szkód:

* - watekZapisPierwotny(): Zapis szkód pożarów pierwotnych do CSV.
* - watekZapisRozprz(): Zapis szkód pożarów rozprzestrzenionych do CSV.

Struktury danych przechowujące szkody:

* - out\_brutto\_final: Wartości szkód brutto.
* - szkody\_katastroficzne: Szkody katastroficzne.
* - wielkosc\_pozaru: Wartości szkód pożaru.

## Format CSV dla Szkód

Pliki CSV zawierają następujące kolumny:

* - Insurer, Longitude, Latitude, Region, Month, SumValue, IndexTable, FireSize, ReasonFire
* - Promien, lat, lon, insurance, resurance, SumValue, WIelkoscKwota, IndexTable, Region, Month, ReasonFire