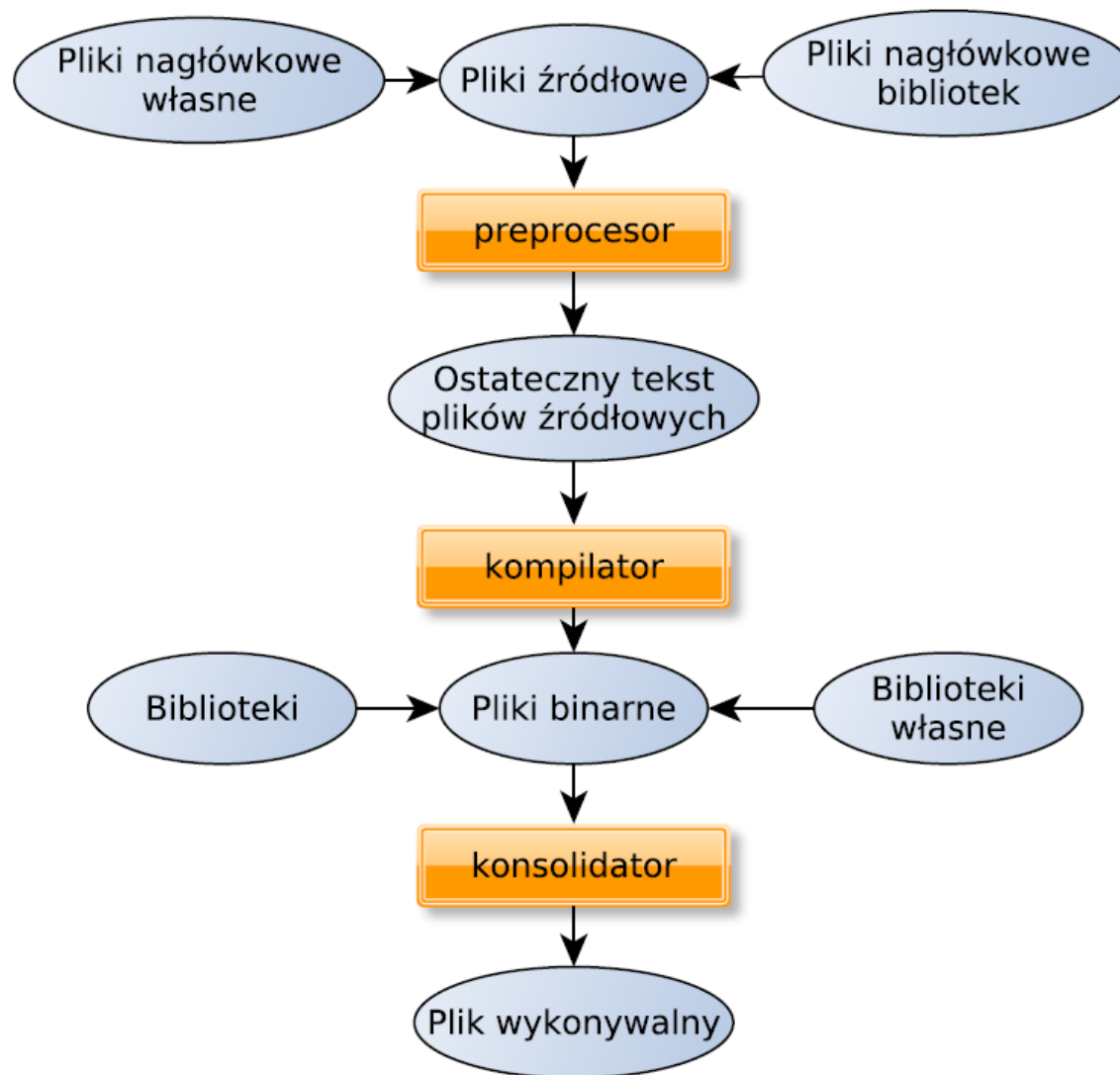
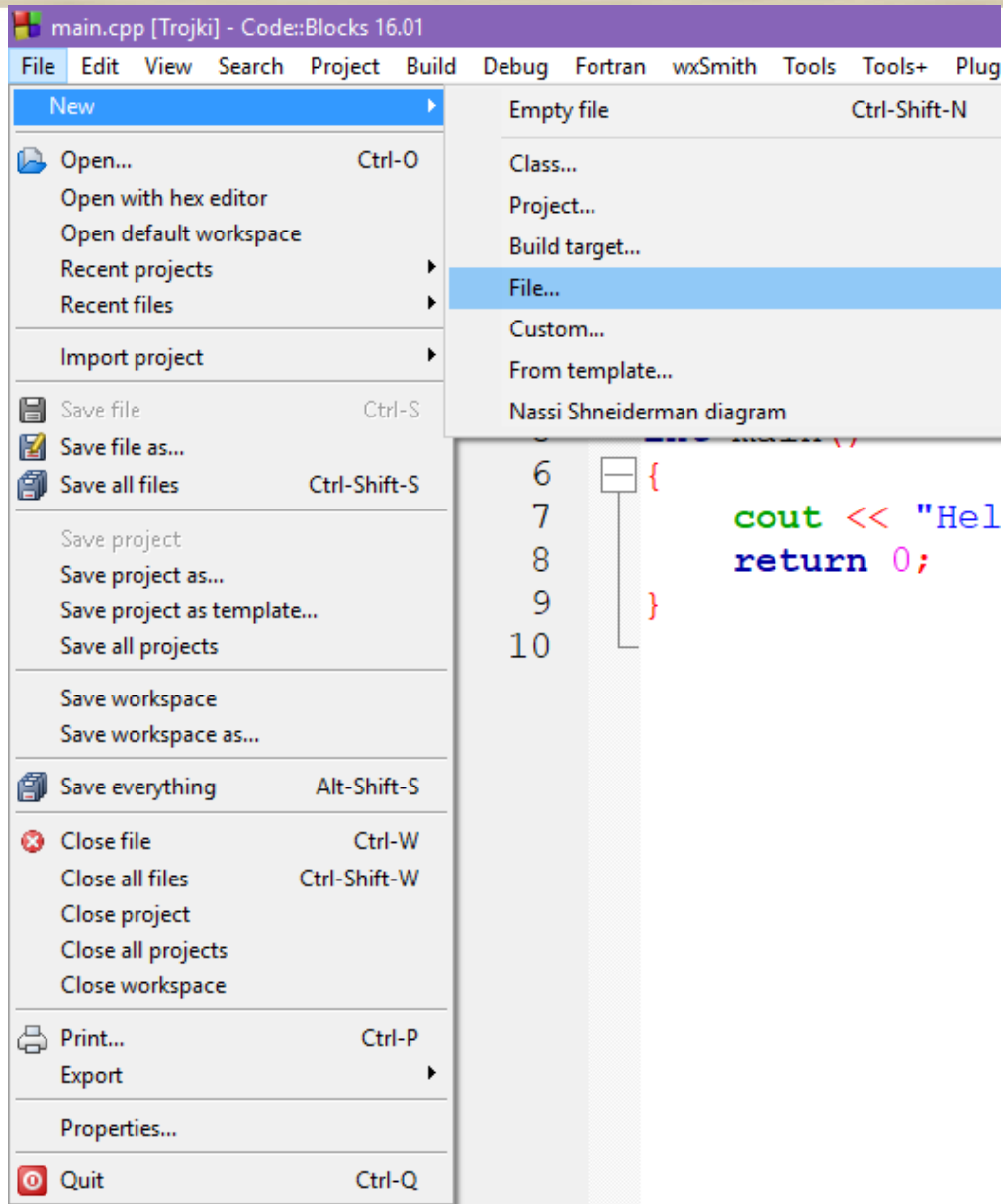


Podział klasy na pliki I metoda

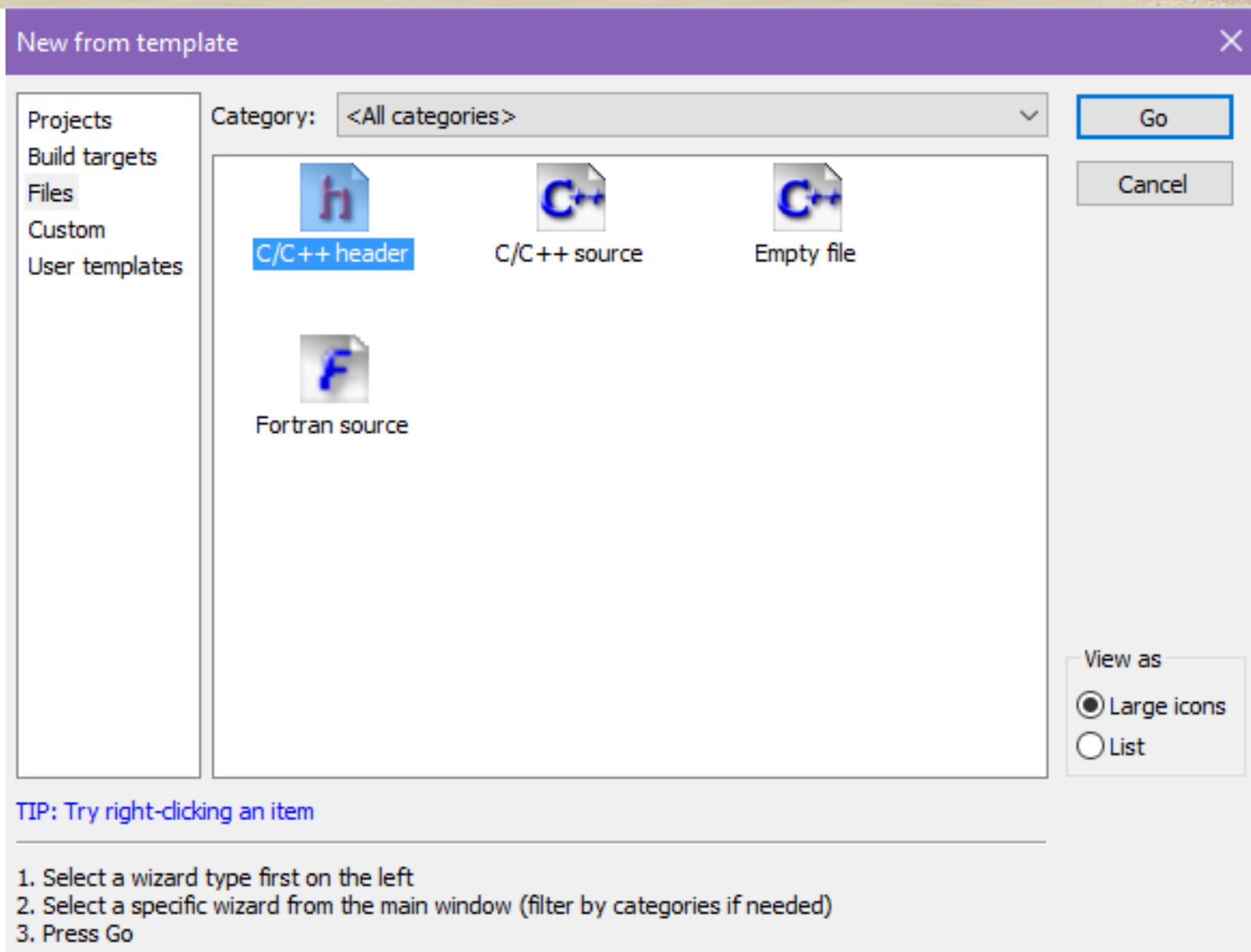
Proces kompilacji



File -> New File



Wybieramy C/C++ header



Wybieramy next



C/C++ header



H/H++ FILE

Welcome to the new C/C++ header file wizard!
This wizard will guide you to create a new C/C++ header file.

When you're ready to proceed, please click "Next"...

☐ Skip this page next time



< Back


Next >

Cancel

Musimy podać ścieżkę i nazwać plik



C/C++ header

 **H/H++ FILE**

Please enter the file's location and name and whether to add it to the active project.

Filename with full path:
 ...

Header guard word:

☒ Add file to active project
In build target(s):

☐ Debug
☐ Release

All None

< Back Finish Cancel

Nazywamy plik z rozszerzeniem .h



Nasz plik jest plikiem nagłówkowym, więc zapisujemy go z odpowiednim rozszerzeniem *nazwa.h*

Nazwa pliku: `funkcje.h`

Zapisz jako typ: `C/C++ header files (*.h;*.hpp;*.hxx;*.hh)`

^ Ukryj foldery

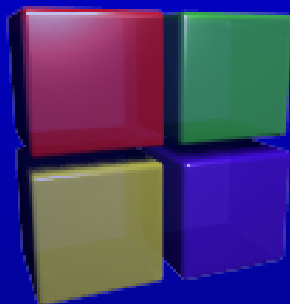
Zaznaczamy obie opcje i FINISH



C/C++ header



H/H++ FILE



Please enter the file's location and name and whether to add it to the active project.

Filename with full path:

C:\Users\yberd\Desktop\Trojki\funkcje.h



Header guard word:

FUNKCJE_H_INCLUDED

☒ Add file to active project
In build target(s):

☒ Debug

☒ Release

All

None

< Back

Finish

Cancel

W pliku nagłówkowym widzimy taki zapis

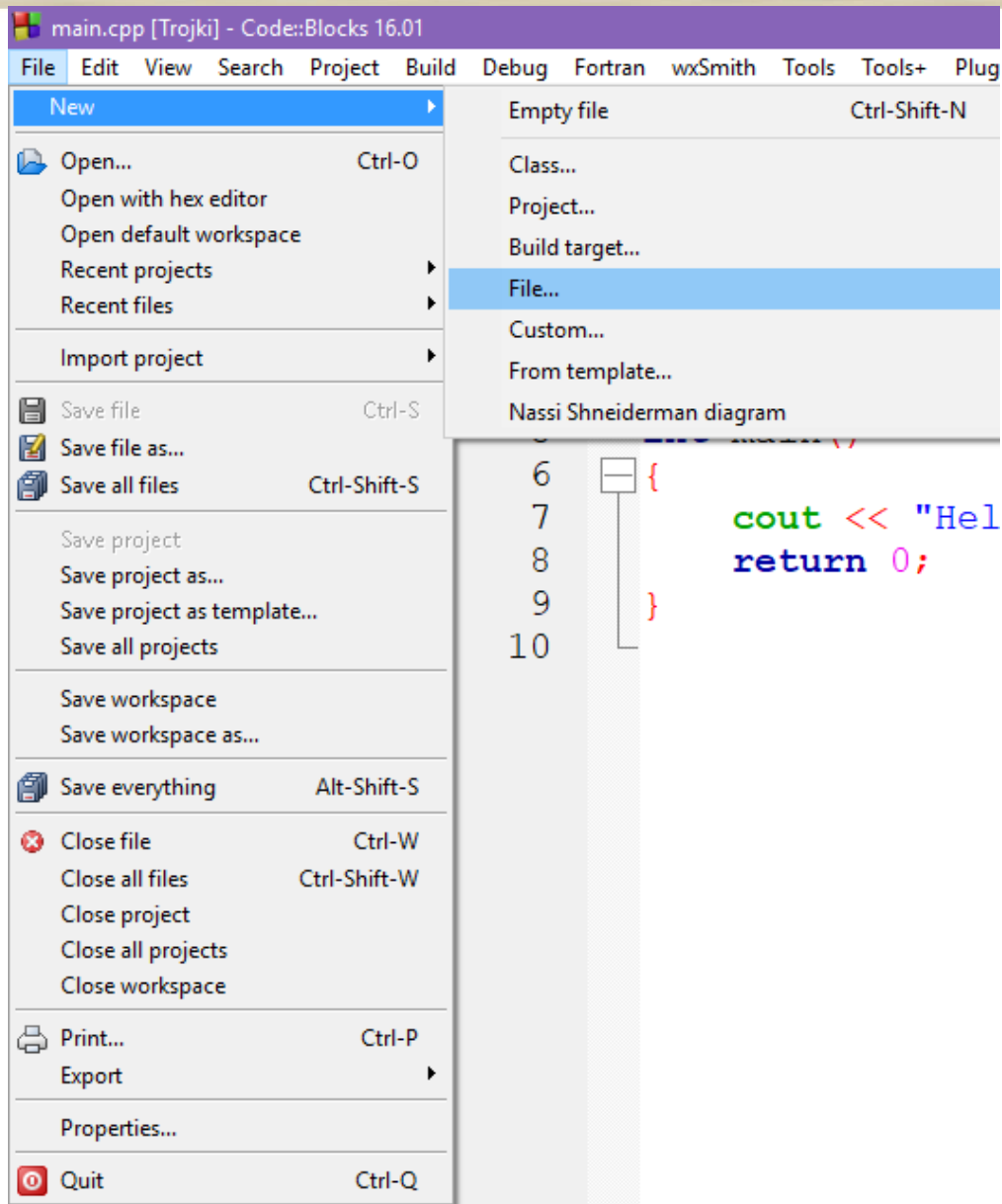


```
1 #ifndef FUNKCJE_H_INCLUDED
2 #define FUNKCJE_H_INCLUDED
```

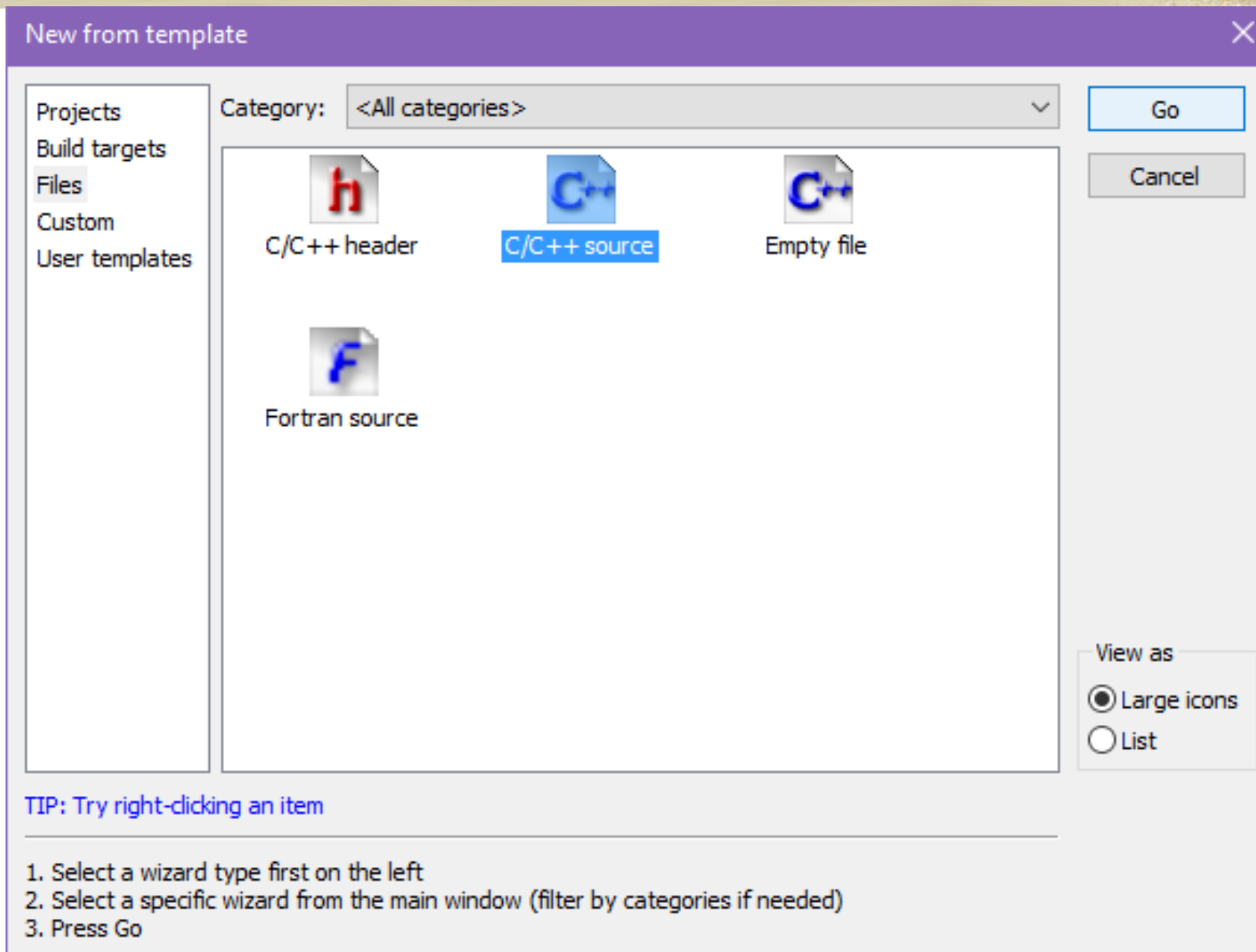
Tu umieścimy nagłówki naszych funkcji

```
3
4
5
6 #endif // FUNKCJE_H_INCLUDED
7
```

Ponownie File -> New File



Tworzymy nowy plik źródłowy



Wybieramy next



C/C++ source



C/C++ FILE

Welcome to the new C/C++ source file wizard!
This wizard will guide you to create a new C/C++ source file.

When you're ready to proceed, please click "Next"...

☐ Skip this page next time



< Back

Next >

Cancel

Jako język wybieramy C++



C/C++ source

C/C++ FILE

Please select the language for the file.

Please make a selection

C
C++

< Back Next > Cancel

Musimy podać ścieżkę i nazwać plik



C/C++ source

C/C++ FILE

Please enter the file's location and name and whether to add it to the active project.

Filename with full path:

...

☒ Add file to active project
In build target(s):

☐ Debug
☐ Release

All None

< Back Finish Cancel

Nazywamy plik z rozszerzeniem .cpp



Nazwa pliku źródłowego (do naszego pliku nagłówkowego) musi być TAKA SAMA, ale z rozszerzeniem .cpp

Nazwa pliku: `funkcje.cpp`


Zapisz jako typ: `C++ files (*.cpp;*.cxx;*.cc)`

^ Ukryj foldery

Zaznaczamy wszystko i finish



C/C++ source

 **C/C++ FILE**

Please enter the file's location and name and whether to add it to the active project.

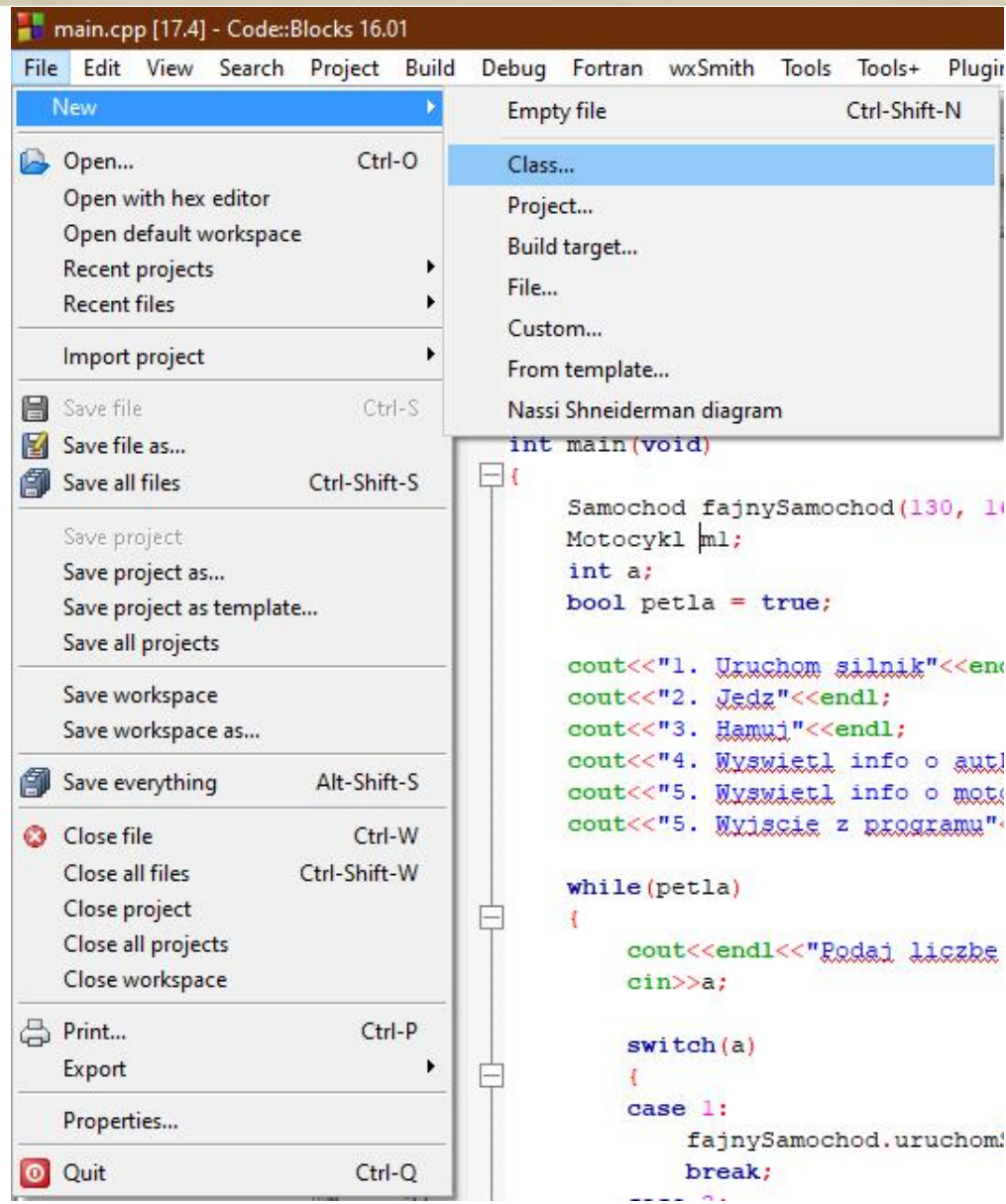
Filename with full path:
 ...

☒ Add file to active project
In build target(s):

- ☒ Debug
- ☒ Release

Podział klasy na pliki II metoda

File -> new class..



Parametry tworzenia klasy



Create new class

Class definition

Class name:

Arguments:

☒ Has destructor ☐ Has copy ctor
☒ Virtual destructor ☐ Has assignment op.

Inheritance

☐ Inherits another class

Ancestor:

Ancestor's include filename:

Scope:

Member variables

Add new:

Scope:

☒ Add "Getter" method
☒ Add "Setter" method
☒ Remove prefix:

Documentation

☐ Add documentation where appropriate

File policy

☒ Add paths to project ☒ Use relative path
☐ Header and implementation file shall be in same folder

Folder:

☐ Header and implementation file shall always be lower case

Header file

Folder: ...

Filename:

☒ Add guard block in header file

Guard block:

Implementation file

☒ Generate implementation file

Folder: ...

Filename:

Header include:

Podział klasy na pliki źródłowe



Plik main.cpp

```
#include <iostream>
#include "Potwor.h"

int main()
{
    Potwor P1;

    P1.obliczAtakFizyczny();
    P1.obliczAtakMagiczny();
    P1.obliczPancerzFizyczny();
    P1.obliczPancerzMagiczny();

    return 0;
}
```

Plik Potwor.h

```
#include <iostream>

class Potwor
{
public:
    std::string nazwa;
    int atakFizyczny;
    int atakMagiczny;
    int pancerzFizyczny;
    int pancerzMagiczny;

    Potwor(std::string="Potwor 1",
           int=60, int=3, int=56, int=2);

    int obliczAtakFizyczny();
    int obliczAtakMagiczny();
    int obliczPancerzFizyczny();
    int obliczPancerzMagiczny();
};
```

Podział klasy na pliki źródłowe cd.



Plik Potwor.cpp

```
#include <iostream>
#include "Potwor.h"
```

```
using namespace std;
```

```
Potwor:: Potwor(string nz, int aF,
                int aM, int pF, int pM)
```

```
{
    nazwa = nz;
    atakFizyczny = aF;
    atakMagiczny = aM;
    pancerzFizyczny = pF;
    pancerzMagiczny = pM;
}
```

```
Potwor:: ~Potwor() {}
```

```
int Potwor:: obliczAtakFizyczny()
{
    //IMPLEMENTACJA
}
```

```
int Potwor:: obliczAtakMagiczny()
{
    //IMPLEMENTACJA
}
```

```
int Potwor:: obliczPancerzFizyczny()
{
    //IMPLEMENTACJA
}
```

```
int Potwor:: obliczPancerzMagiczny()
{
    //IMPLEMENTACJA
}
```