POPRO – lab 1

Autor: Łukasz Dąbała

1 Tematyka

Laboratorium 1 z przedmiotu POPRO, dotyczy zintegrowanego środowiska programistycznego oraz debuggera.

2 Przygotowanie do zajęć laboratoryjnych

W ramach przygotowania do laboratorium warto zapoznać się z materiałami udostępnianymi na stronie Visual Studio Code – `https://code.visualstudio.com/docs/cpp/config-mingw`.

3 Przebieg laboratorium

W trakcie laboratorium należy wykonać wszystkie ćwiczenia. Po ich skończeniu należy wysłać spakowane pliki źródłowe oraz odpowiedzi na pytania na maila prowadzącego: **Lukasz.Dabala@pw.edu.pl**. Polecenia od 3.3 należy wykonać w zintegrowanym środowisku programistycznym – Visual Studio Code.

3.1 MinGW

W przypadku korzystania z systemu Windows należy zainstalować kompilator MinGW - `https://sourceforge.net/projects/mingw-w64/files/Toolchains%20targetting%20Win32/Personal%20Builds/mingw-builds/installer/mingw-w64-install.exe/download` Aby możliwe było użycie go z konsoli, czy później z docelowego IDE, należy dodać go do zmiennej środowiskowej PATH. Do tej zmiennej należy dodać ścieżkę do folderu *bin* z folderu instalacji kompilatora. Po instalacji należy sprawdzić, czy można kompilator uruchomić z konsoli. W tym celu otwieramy konsolę (*cmd* lub *powershell*) i wpisujemy:

```
gcc --version
```

Jeżeli nie ma żadnego błędu można przejść do kolejnego punktu.

3.2 Kompilacja i uruchomienie prostego programu w języku C

W ramach ćwiczenia należy skompilować oraz uruchomić prosty program w języku C przy użyciu terminala. W tym celu należy stworzyć plik *simple.c* oraz wkleić do niego poniższy kod.

```
#include <stdio.h>

int main() {
    printf("Hello world");
}
```

Następnie w konsoli, w katalogu, w którym znajduje się nowo utworzony plik należy wpisać polecenie:

```
gcc simple.c -o simple.exe
```

Polecenie skompiluje podany plik i utworzy plik wykonywalny. W celu wykonania programu należy wykonać:

```
.\simple.exe
```

W ramach tego ćwiczenia należy zmodyfikować program tak, aby wypisywał login użytkownika.

3.3 Tworzenie projektu i uruchamianie

W celu stworzenia nowego projektu na dysku należy stworzyć folder np. *SimpleProgram*, w którym docelowo będziemy trzymać pliki źródłowe. Następnie dany folder otwieramy w programie Visual Studio Code przez opcję *File - Open Folder*. Dodajmy nasz plik źródłowy, który będziemy modyfikować: *File - New File*. Zapisujemy plik pod nazwą *simple.c*. W samym pliku dopisujemy podobnie jak w poprzednim ćwiczeniu kod:

```
#include <stdio.h>

int main() {
    printf("Hello world");
}
```

Nasz najprostszy program jest gotowy.

Aby uprościć sobie pracę jak i umożliwić sobie kompilację programu bezpośrednio ze środowiska należy zainstalować dodatkowe wtyczki:

1. C/C++
2. C/C++ Compile Run

Projekt „NERW 2 PW. Nauka – Edukacja – Rozwój – Współpraca” współfinansowany jest ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego

Fundusze Europejskie Wiedza Edukacja Rozwój
Rzeczpospolita Polska
Politechnika Warszawska
Unia Europejska Europejski Fundusz Społeczny

W celu instalacji wtyczek wybieramy *View - Extensions*.

Aby zbudować program musimy stworzyć konfigurację zadania. W tym celu wybieramy: *Terminal - Configure Tasks... - gcc for active file*. Po stworzeniu konfiguracji możemy zbudować nasz program: *Terminal - Run Build Task...*.

W terminalu spróbujmy następnie uruchomić nasz program wpisując jak w poprzednim ćwiczeniu:

```
.\simple.exe
```

3.4 Debuggowanie

W ramach ćwiczenia, trzeba prześledzić wykonanie programu z wykorzystaniem debuggera oraz poprawić błędy w programie. Program wczytuje od użytkownika zmienne, a następnie liczy ich średnią.

```c
#include <stdio.h>
#include <stdlib.h>

int main(int argc, char ** argv) {
    int sum = 0;
    for (int i = 0; i < argc; ++i) {
        sum += atoi(argv[i]);
    }

    int average = sum / (argc);
    printf("Average: %d", average);
}
```

Należy prześledzić wykonanie pętli parsującej argumenty oraz liczącej średnią. W tym celu warto postawić pułapki (breakpoint) w linii dodającej nowe wartości do tablicy oraz w linii liczącej średnią. Dzięki temu w czasie wykonania program zatrzyma się w wyznaczonych miejscach oraz umożliwi podejrzenie zmiennych.

Pytania:

1. Czy dany program zawsze liczy poprawnie średnią?
2. Co należy poprawić, aby obliczenia były poprawne?
3. Czy należy też poprawić wypisywanie wyniku?

3.5 Komentowanie kodu

W języku C należy komentować pliki, funkcje oraz trudne fragmenty kodu. W czasie trwania laboratorium należy dopisać komentarze do fragmentu kodu wykorzystanego w poprzednim ćwiczeniu. W tym celu należy dopisać komentarz do każdej linii wyjaśniający jej działanie. Typy komentarzy:

Projekt „NERW 2 PW. Nauka – Edukacja – Rozwój – Współpraca" współfinansowany jest ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego

Fundusze Europejskie Wiedza Edukacja Rozwój
Rzeczpospolita Polska
Politechnika Warszawska
Unia Europejska Europejski Fundusz Społeczny

```
// line comment

/*
Block comment
*/

/**
Function documentation comment
@param describes a parameter
@return describes the return value
*/
```

Prowadzącemu należy przesłać skomentowany kod.

Projekt „NERW 2 PW. Nauka – Edukacja – Rozwój – Współpraca” współfinansowany jest
ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego