Rozdział 1

Wprowadzenie

Niniejszy skrypt dedykowany jest studentom pierwszego roku inżynierii medycznej, którzy nie mieli wcześniej kontaktu z żadnym z języków programowania. W pierwszych rozdziałach student zaznajomi się od strony praktycznej z rodzajami algorytmów oraz krótkim specjalnie okrojonym wprowadzeniem do programowania dla inżynierów...

Ocenie z laboratorium podlega projekt do samodzielnego wykonania. Projekty można podzielić ze względu na maksymalne oceny jakie można dostać za ich wykonanie. Brak projektów dwuosobowych. Podstawową umiejętnością przydatną w programowaniu jest czytanie ze zrozumieniem komunikatów oraz przeszukiwanie pomocy w celu znalezienia odpowiedzi na pytanie.

1. Projektowanie programu i algorytmy

Wstęp do programowania dla inżynierii medycznej przewiduje krótkie wprowadzenie do projektowania algorytmów oraz ich zapisu i interpretacji w postaci diagramowej.

- Co to jest algorytm?
- Schemat decyzyjny, bloki wejścia/wyjścia, bloki warunkowe, instrukcje
- Proste przykłady: NWD, NWW, konwersja liczby (na dowolny system, np. binarny)

Po tym rozdziale student wie jak zaprojektować proste algorytmy.

...

2. Wybrany język programowania

Skrypt nie zakłada znajomości żadnego języka programowania. Ze względów na bardzo szerokie zastowowania, wysoką wydajność oraz możliwości wybrano język *C*.

2.1. Instalacja kompilatora C / C++

Większość instalacji będzie zawierała kompilator języka C (gcc) oraz C++ (g++). W zależności od używanego systemu operacyjnego w skrótowy sposób opisano kilka możliwych scenariuszy. Większość studentów korzysta z systemu operacyjnego `Microsoft Windows 10` z tego powodu bardziej szczegółowy opis instalacji znajduje się w pod koniec tego podrozdziału. Wybrane sceneriusze przygotowania środowiska programistycznego:

1. Linux: kompilator G++ 4.9.2 lub nowszy(notatnik + kompilator + prosty Makefile ze względów praktycznych)

   Polecenie 1.1: Instalacja kompilatora g++ na dystrybucji Debian'a, Ubuntu i pochodnych

   ```
   sudo apt install g++
   ```

   alternatywnie można zainstalować pakiet: `sudo apt install build-essentials`, który zawiera m. in. `g++`.

   Polecenie 1.2: Instalacja kompilatora g++ na dystrybucji Red Hat, Fedora, CentOS

   ```
   sudo yum install g++
   ```

   lub w nowszych wersjach

   Polecenie 1.3: Instalacja kompilatora g++ na dystrybucji Red Hat $\geq$ 7.x

   ```
   sudo dnf install g++
   ```

   Jeżeli student posiada zainstalowaną inną dystrybucję linux'a, np. Arch czy Slackware z pewnością nie potrzebuje pomocy z instalacją kompilatora.

2. Linux: kompilator G++ 4.9.2 lub nowszy(CodeBlocks)
3. Mac OS: z Apple Store należy zainstalować XCode z pakietem g++
4. MS Windows: podsystem Ubuntu: instalacja G++ jak w zwykłym ubuntu. **Uwaga**: zainstalowanie CodeBlocks w połączeniu z Xming na podsystemie jest możliwe, ale nie zalecane.
5. MS Windows: CodeBlocks + MinGW bez instalacji podsystemu linux
6. **MS Windows 10**: Notatnik++ w połączeniu z MinGW64 (gcc 7.2.0) oraz prosty Makefile
   Ten scenariusz zawiera dokładne wersje oprogramowania zainstalowane w pracownii komputerowej **V12**.

   Pobieranie wymaganych składników

   (a) Kompresja danych 7zip (1801, 64-bit): pobierz
   (b) Kompilator C/C++ (x86_64, 7.2.0, seh): pobierz

(c) Notatnik++ (7.5.4, 64-bit): pobierz

(d) Wtyczka NppExec do programu Notatnik++ (v0.6 alpha 1, 64bit) pobierz

Krótka instrukcja instalacji

(a) Instalacja programu Notatnik++

(b) Do folderu w którym zainstalowany jest Notatnik++/plugins należy wkleić plik `NppExec.dll`

(c) Aby ustawienia wtyczki `NppExec` były zapisywane należy utworzyć katalog dla użytkownika

Polecenie 1.4: Utworzenie katalogu dla ustawień wtyczek programu Notatnik++. Polecenie można wywołać z poziomu wiersza poleceń (`cmd`)

```
mkdir C:\Users\Student\
  AppData\Roaming\Notatnik++\plugins\config
```

gdzie katalog `\Users\Student` należy podmienić katalogiem domowym swojego użytkownika

(d) Instalacja programu 7zip

(e) Rozpakować kompilator do katalogu `C:\Dodatki\mingw-seh`, tak aby dostępny był program `C:\dodatki\mingw-seh\bin\mingw32-make`

(f) Dodanie ścieżki kompilatora do zmiennej środowiskowej `PATH`. Ten krok jest potrzebny, żeby polecenie kompilatora `gcc` oraz `mingw32-make` można wywołać z dowolnego katalogu bez podawania bezwzględnej ścieżki do programu: `C:\Dodatki\mingw-seh\bin\gcc`. **Uwaga:** poniższa instrukcja dodawania może wywołać trwałe zmiany w systemie operacyjnym i należy ją zastosować ze szczególną ostrożnością i wyłącznie na własną odpowiedzialność:

Listing 1.5: Dodawanie ścieżki kompilatora do PATH. Polecenie można wywołać z poziomu PowerShell dostępnego w systemie MS Windows 10.

```
$Env:Path+= "C:\Dodatki\mingw_seh\bin\;";
[Environment]::SetEnvironmentVariable("Path",
$env:Path,[System.EnvironmentVariableTarget]
::User)
```

2.2. Skrypt do wtyczki NppExec

```
NPP_SAVE // zapisuje aktualnie edytowany plik

// przejscie do katalogu z plikiem
CD $(CURRENT_DIRECTORY)

// wykonanie polecenia make
mingw32-make ename=$(NAME_PART) -f Makefile.win

```

2.3. Kod źródłowy `pierwszy.c`

Listing 1.6: Kod źródłowy programu wyświetlającego tekst na ekranie.

```
#include <stdio.h> // naglowek std. input/output
#include <stdlib.h> // naglowek std. library

int main()
{
  printf("Witaj swiecie"); //
  system("pause"); // polecenie systemowe: pause
  return 0;
}

```

2.4. Makefile.win

Pliki `Makefile` służą m. in. do zautomatyzowania kompilacji kodu źródłowego i jego zależności. Na pierwszych zajęciach program składa się tylko z jednego pliku *.c jednak praca z plikami `Makefile` pozwala uniezależnić się od dużych środowisk programistycznych i pracować nad tym samym kodem źródłowym w różnych systemach operacyjnych podmieniając jedynie plik `Makefile`.

Listing 1.7: Definicja pliku Makefile.win

```
APPNAME=pierwszy

build:
    gcc $(APPNAME).c -o $(APPNAME).exe
```

Dla programów, których kod źródłowy zawarty jest w jednym pliku można skorzystać z alternatywnej definicji pliku `Makefile.win`, a nazwę tego pliku można podać wywołując polecenie `mingw32-make`, tak jak np. w skrypcie NppExec.

Listing 1.8: Alternatywna definicja pliku Makefile.win wykorzystana do pracy z wtyczką NppExec.

```
build:
    gcc $(ename).c -o $(ename).exe
```

Uwaga: w systemach operacyjnych Linux/Mac OS zamiast polecenia `mingw32-make` należy używać `make`.