

# StudentMachine

Instrukcja obsługi

[https://github.com/marcin-filipiak/cpp\\_studentmachine](https://github.com/marcin-filipiak/cpp_studentmachine)

1.04.2024  
Marcin Filipiak

# Wstęp

StudentMachine jest aplikacją przeznaczoną dla uczniów, wspomaga pracę na zajęciach z przedmiotów; „podstawy programowania”, „aplikacje internetowe” itp.

Zadaniem aplikacji jest instalacja oprogramowania, konfiguracja systemu operacyjnego, przygotowanie szablonów z projektami, pobieranie ćwiczeń a także po zalogowaniu ucznia pobranie jego prac z Github i ich automatyczne zapisanie.

Apliacja przeznaczona jest dla systemów Linux; Debian, Ubuntu.

Dostępna jest również wirtualna maszyna dla Oracle Virtual Box, skonfigurowana i gotowa do uruchomienia na zajęciach. Aktualny link do maszyny znajduje się na stronie startowej projektu w dziale „LINKS”: [https://github.com/marcin-filipiak/cpp\\_studentmachine](https://github.com/marcin-filipiak/cpp_studentmachine)

## Spis treści

Wstęp.....	2
1. Instalacja programu w systemie.....	3
1.1. Przygotowanie systemu.....	3
2. Pierwsze uruchomienie wirtualnej maszyny.....	3
3. Rozpoczęcie pracy.....	3
3.1. Założenie repozytorium.....	4
3.2. Rejestracja repozytorium w StudentMachine.....	5
4. Założenie templejtów.....	6
5. Zapisanie pracy.....	6
6. Zakończenie pracy.....	6
7. Rozpoczęcie pracy po raz kolejny.....	6
8. Pobranie ćwiczenia od nauczyciela.....	7
9. Aktualizacja.....	7
10. Załączniki.....	8
10.1 Podstawowe polecenia w Linux.....	8
10.2 Pierwsze ćwiczenie w c++.....	9

# 1. Instalacja programu w systemie

Jeśli nie używasz gotowej wirtualnej maszyny i chcesz zainstalować program w swoim systemie wykonaj następujące kroki w konsoli.

1. Pobierz aplikację:

```
sudo wget https://github.com/marcin-filipiak/cpp_studentmachine/raw/main/client/build/studentmachine -P /bin
```

2. Nadaj jej prawa do wykonywania:

```
sudo chmod +x /bin/studentmachine
```

## 1.1. Przygotowanie systemu

Po dokonaniu instalacji programu skonfiguruj system, zostaną zainstalowane niezbędne kompilatory i narzędzia.

```
studentmachine install
```

## 2. Pierwsze uruchomienie wirtualnej maszyny

1. Zainstaluj Oracle Virtual Box ze strony <https://www.virtualbox.org/wiki/Downloads>

2. Pobierz maszynę do której link znajdziesz na stronie projektu w dziale „LINKS”:  
[https://github.com/marcin-filipiak/cpp\\_studentmachine/](https://github.com/marcin-filipiak/cpp_studentmachine/)

3. Kliknij dwa razy na pobrany plik z wirtualną maszyną i zaimportuj ją.

4. Uruchom wirtualną maszynę w Virtual Box

5. Zaloguj się z loginem: uczen, hasło: internet

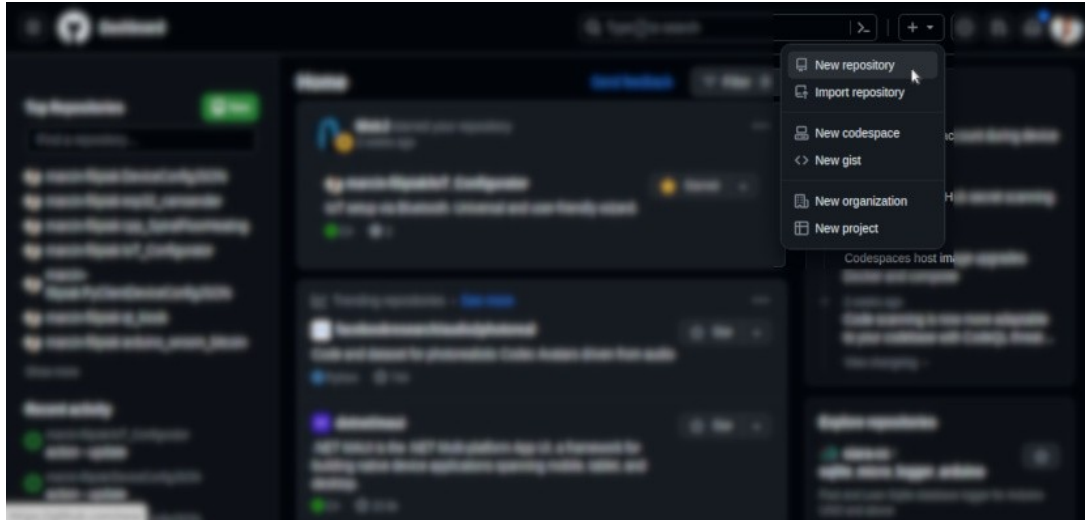
## 3. Rozpoczęcie pracy

Program studentmachine i instrukcje dla niego wywoływane są w konsoli.

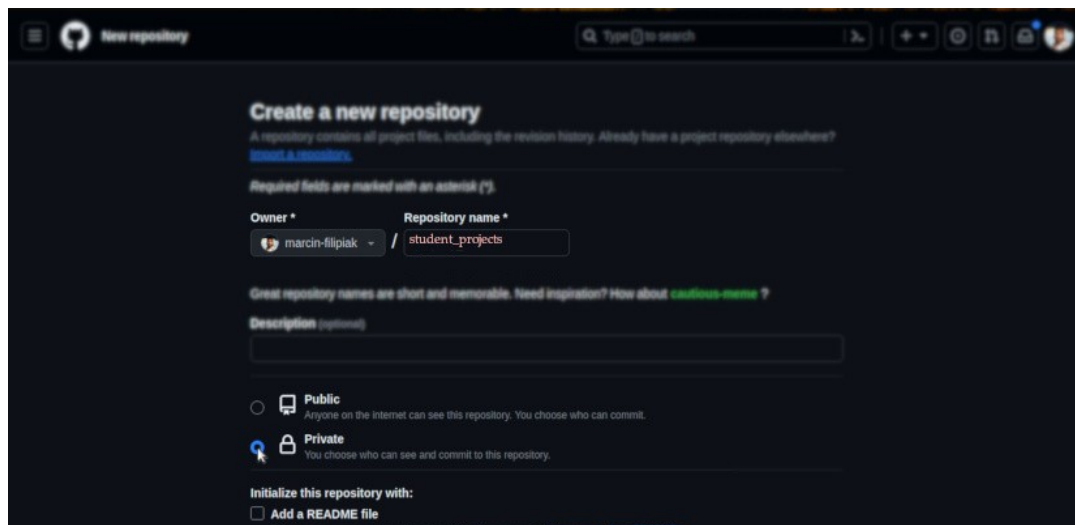
Praca może zostać wykonana w dwóch trybach; synchronizacji plików z Github i bez synchronizacji. Zaleca się pracę z synchronizacją plików, wykonane ćwiczenia będą wtedy przechowywane w repozytorium Github i dostępne przez sieć z dowolnego miejsca i na dowolnym komputerze. Brak synchronizacji spowoduje wykasowanie prac przez następnego ucznia pracującego w systemie.

## 3.1. Założenie repozytorium

Aby przechowywać swoje prace w serwisie <https://github.com> należy posiadać konto. Zaloguj się i załóż nowe repozytorium.



Jako nazwę repozytorium podaj obowiązkowo: `student_projects`



Zaznacz, czy chcesz by było ono dostępne dla innych czy też nie, możesz dodać plik README.

## 3.2. Rejestracja repozytorium w StudentMachine

Połączenie z github wykonywane jest za pomocą kluczy. Program StudentMachine wygeneruje je dla Ciebie i połączy się z repozytorium. Upewnij się, że wykonałeś wszystko zgodnie z punktem 3.1. Repozytorium będzie synchronizowane z folderem „student\_projects” w twoim katalogu domowym.

Program dokona wygenerowania pary kluczy. Klucz publiczny dodamy do Github, natomiast zabezpieczony hasłem klucz prywatny będzie przechowywany na serwerze kluczy StudentMachine i zapewni ci dostęp do twoich plików na dowolnym komputerze z tym programem.

Wykonaj te polecenia w przypadku gdy uruchamiasz StudentMachine po raz pierwszy lub nie pamiętasz hasła do klucza.

1. Uruchom w konsoli:

*studentmachine systemup*

2. Wybierz opcję „register” klikając na klawiaturze „r” i potwierdzając enterem.

3. Podaj swój login na Github, a następnie email

4. Program generuje klucz publiczny i prywatny. Podaj hasło (passphrase) która zabezpieczy klucz przed nieautoryzowanym użyciem.

5. Program wysyła klucz prywatny na serwer, by był on dostępny przy następnym uruchomieniu programu.

6. Następuje rejestrowanie klucza w systemie operacyjnym, podaj hasło ucznia: internet

7. Teraz należy zarejestrować klucz publiczny na Github.

Uruchom przeglądarkę internetową i przejdź na serwer kluczy:

<http://api.noweenergie.org/application/StudentMachine/keyring/>

Podaj w formularzu swój login na github, zaznacz i skopiuj klucz publiczny.

8. Upewnij się, że jesteś zalogowanym na stronie <http://github.com> i następnie przejdź pod adres: <https://github.com/settings/keys>

9. Wybierz „New SSH key” i w polu „Key” wklej skopiowany klucz.

Pole „Title” możesz uzupełnić według własnego uznania, „KeyType” ustawiony jako „Authentication Key”.

10. Wróć do konsoli z programem studentmachine. Potwierdź wykonanie rejestracji klucza przyciskiem „y” i zatwierdź enterem.

11. Program utworzy w twoim katalogu domowym folder „student\_project” i zsynchronizuje go z repozytorium.

## 4. Założenie templejtów

Rozpoczynając pracę warto rozpocząć od założenia templejtów. Program automatycznie stworzy szablony projektów w C++ i Python. Doda dokumenty z instrukcjami, a także skrypty kompilujące źródło i uruchamiające program.

Wykonaj polecenie:

```
studentmachine templates
```

## 5. Zapisanie pracy

W dowolnym momencie można wysłać zawartość folderu student\_projects na swój GitHub.

Wykonaj polecenie:

```
studentmachine savework
```

## 6. Zakończenie pracy

Zamknięcie systemu, wysłanie zapisanych prac z folderu student\_projects na twój GitHub, oraz wyczyszczenie maszyny dla następnego ucznia.

Wykonaj polecenie:

```
studentmachine systemdown
```

## 7. Rozpoczęcie pracy po raz kolejny

Jeśli jesteś już zarejestrowanym użytkownikiem (przeprowadziłeś procedurę z pkt. 3.2) i po raz kolejny logujesz się w systemie uruchom studentmachine, by przywrócił ostatnio zapisane przez ciebie prace.

Wykonaj polecenie:

```
studentmachine systemup
```

1. następnie wybierz opcję „u”
2. podaj login z github i twój mail
3. podaj passphrase do klucza  
(jeśli nie pamiętasz passphrase wykonaj rejestrację nowego klucza pkt. 3.2)

## 8. Pobranie ćwiczenia od nauczyciela

Nauczyciel ma możliwość udostępnienia ćwiczeń. Ćwiczenie trafia do folderu `student_project` i może zawierać też skrypty które zostaną wykonane automatycznie, by przygotować twój system do pracy.

Wykonaj polecenie:

```
studentmachine exercise nazwa_cwiczenia
```

gdzie w miejsce „nazwa\_cwiczenia” wpisz podaną przez nauczyciela nazwę.

## 9. Aktualizacja

Aktualizacja pobiera najnowszą wersję programu `studentmachine` i instaluje ją w systemie.

Wykonaj polecenie:

```
studentmachine update
```

## 10. Załączniki

### 10.1 Podstawowe polecenia w Linux

#### 1. ls (List):

- `ls` służy do wyświetlania zawartości bieżącego katalogu.
- Przykłady:
  - `ls` - wyświetli listę plików i katalogów w bieżącym katalogu.
  - `ls -l` - wyświetli szczegółowe informacje o plikach.
  - `ls -a` - pokaże ukryte pliki (te zaczynające się od kropki).

#### 2. cd (Change Directory):

- `cd` służy do zmiany aktualnego katalogu.
- Przykłady:
  - `cd nazwa_katalogu` - zmieni katalog na podany.
  - `cd ..` - przejdzie do katalogu nadrzędnego.

#### 3. rm (Remove):

- `rm` usuwa pliki lub katalogi.
- Przykłady:
  - `rm plik.txt` - usunie plik o nazwie "plik.txt".
  - `rm -r katalog` - usunie katalog wraz z jego zawartością (użyj ostrożnie!).

#### 4. touch:

- `touch` tworzy pusty plik o podanej nazwie.
- Przykład:
  - `touch nowy_plik.txt` - stworzy nowy, pusty plik o nazwie "nowy\_plik.txt".

#### 5. Uruchomienie skryptu sh:

- Aby uruchomić skrypt shell (bash) o rozszerzeniu `.sh`:
  - `sh nazwa_skryptu.sh` - uruchomi skrypt o podanej nazwie.

#### 6. Nano - Edytor Tekstu:

- `nano` to prosty edytor tekstu w terminalu.
- Aby uruchomić Nano: `nano nazwa_pliku`.
- Podstawowe komendy w Nano:
  - `Ctrl + X` - Wyjście z Nano.
  - `Ctrl + O` - Zapisz plik.
  - `Ctrl + W` - Wyszukaj.
  - `Ctrl + G` - Pomoc.



## 10.2 Pierwsze ćwiczenie w c++

Jeśli posiadasz już skonfigurowany system (wykonałeś punkty 1, 2, 3, 4) i wykonałeś poprawnie – rozpocznij nowe ćwiczenie. Polecenia Linux znajdziesz w punkcie 10.1.

Wchodzimy do folderu gdzie przechowywane są prace:

```
cd ~/student_projects
```

Sprawdź poniższym poleceniem czy widzisz folder cpp (jeśli nie, wykonaj punkt 4):

```
ls
```

Wchodzimy do folderu cpp:

```
cd cpp
```

Kopiujemy szablon projektu do folderu cwiczenie1:

```
cp -r szablon cwiczenie1
```

Wchodzimy do folderu cwiczenie1:

```
cd cwiczenie1
```

Rozpoczynamy edycję kodu (nano lub mcedit) c++:

```
nano main.cpp
```

By zapisać pracę wciśnij CTR+X, zwróć uwagę na komunikaty w dolnej części ekranu

Rozpocznij kompilację:

```
./kompiluj
```

Jeśli pojawiły się błędy wróć do edycji kodu i je popraw. Jeśli nie, masz gotowy program i możesz go uruchomić poleceniem:

```
./main
```