

Wstęp do programowania

Wykład 1. Podstawy C.

Wstęp do programowania

Michał Cieśla

**pokój D-2-47
konsultacje: piątek 10-12**

michal.ciesla@uj.edu.pl

Zagadnienia:

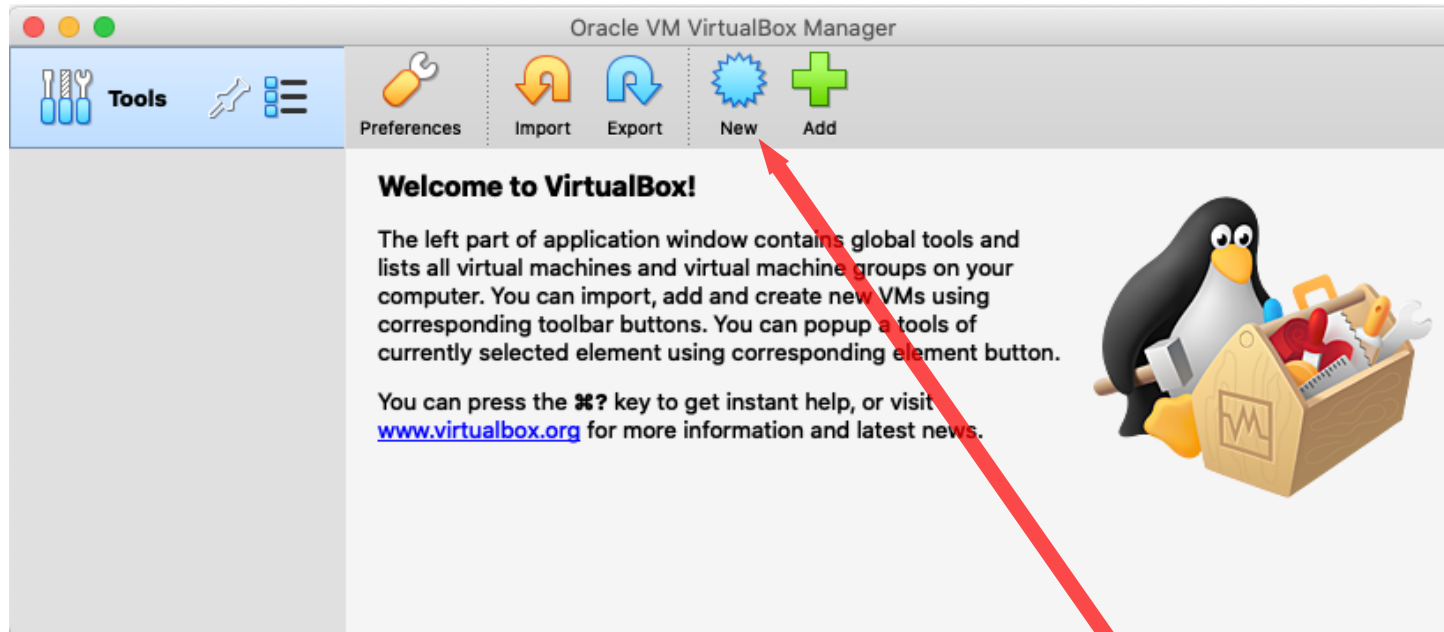
- programowanie proceduralne: C,**
- programowanie obiektowe: C++/Java/Python,**
- podstawowe struktury danych i algorytmy,**
- programowanie funkcyjne: Scala,**
- programowanie współbieżne.**

Plan wykładu

Pierwszy program w C:

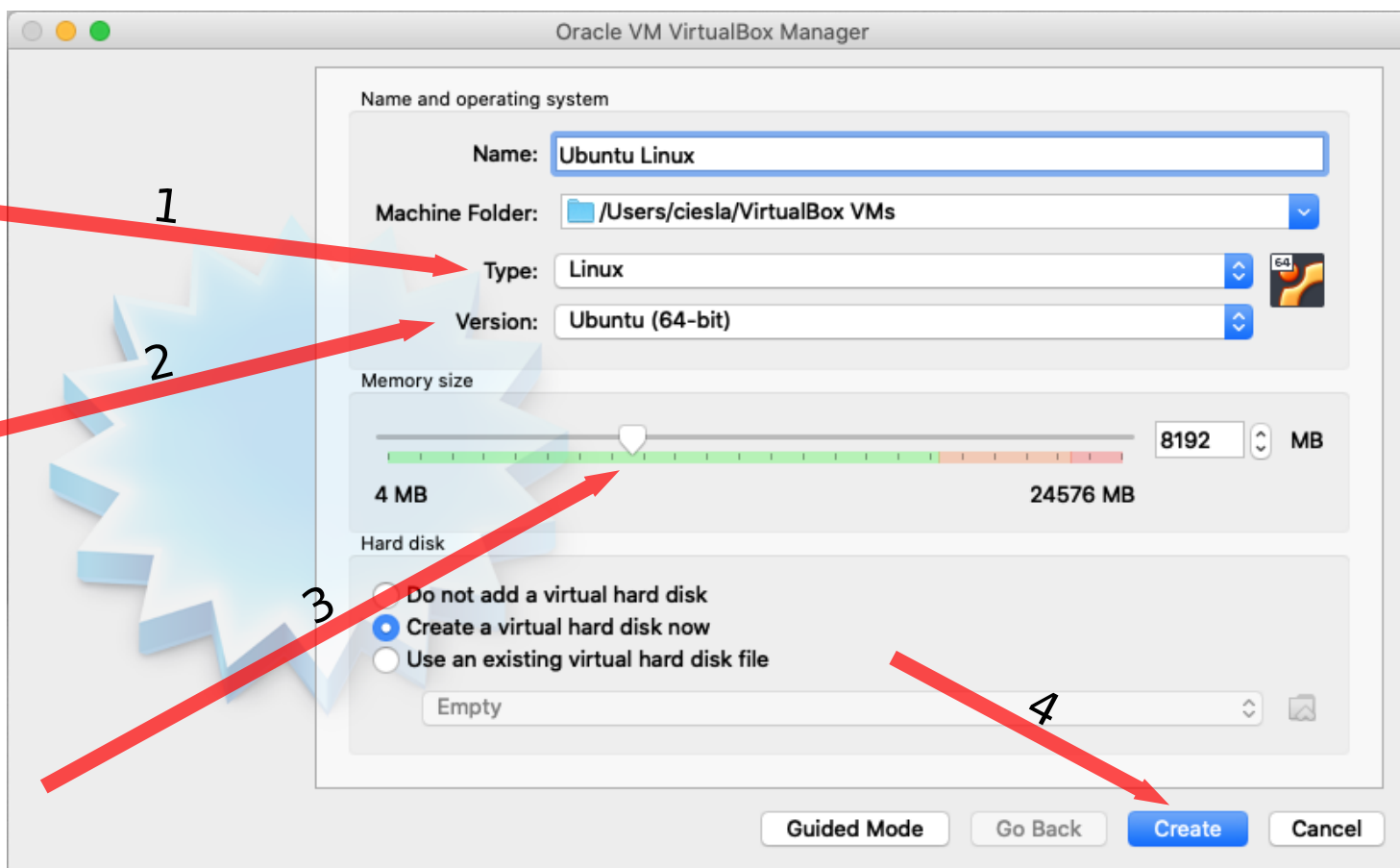
- **Linux / VirtualBox,**
- **kompilacja i uruchomienie,**
- **funkcje w C,**
- **podstawowe typy danych.**

VirtualBox



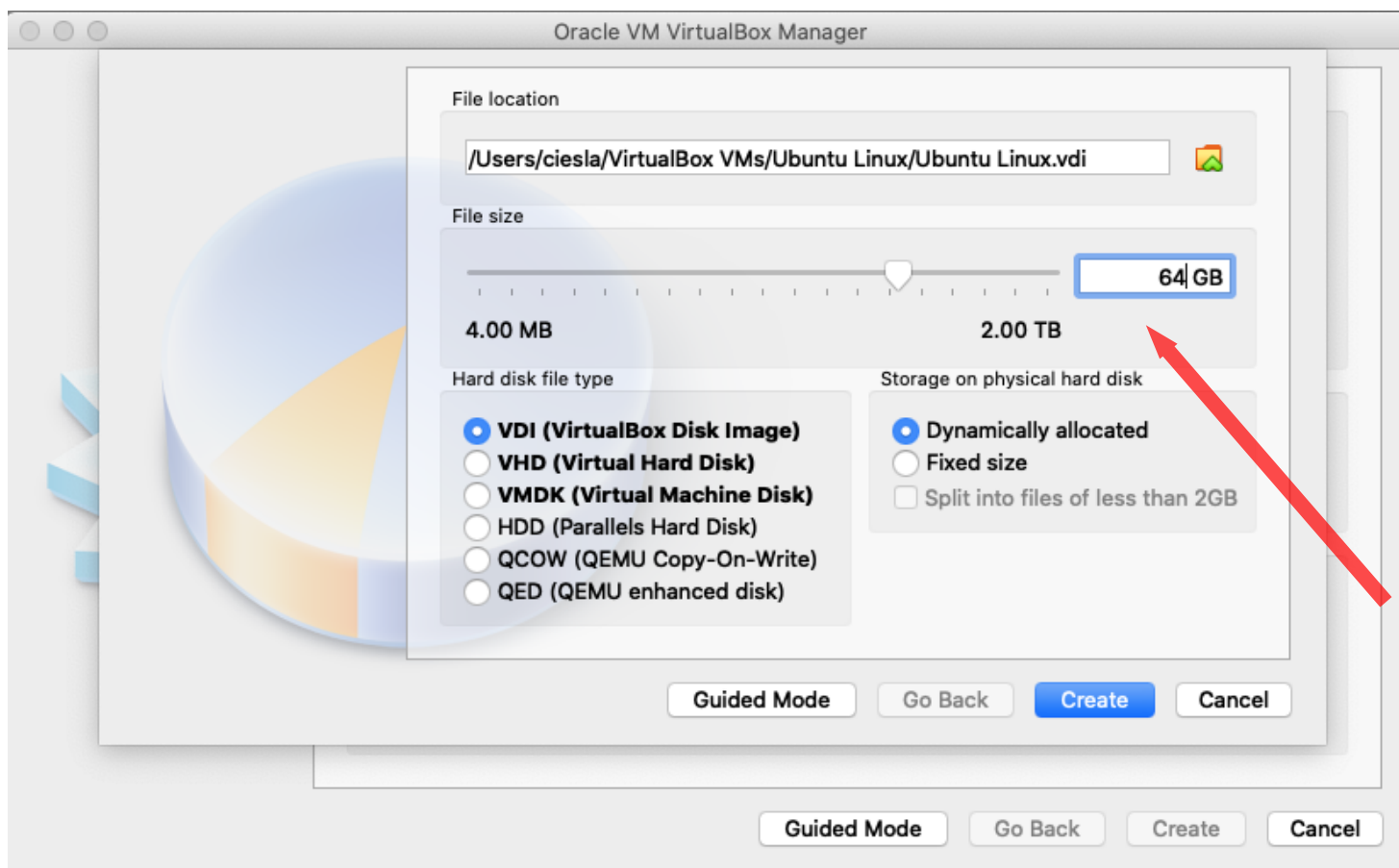
Klikamy „New”

VirtualBox



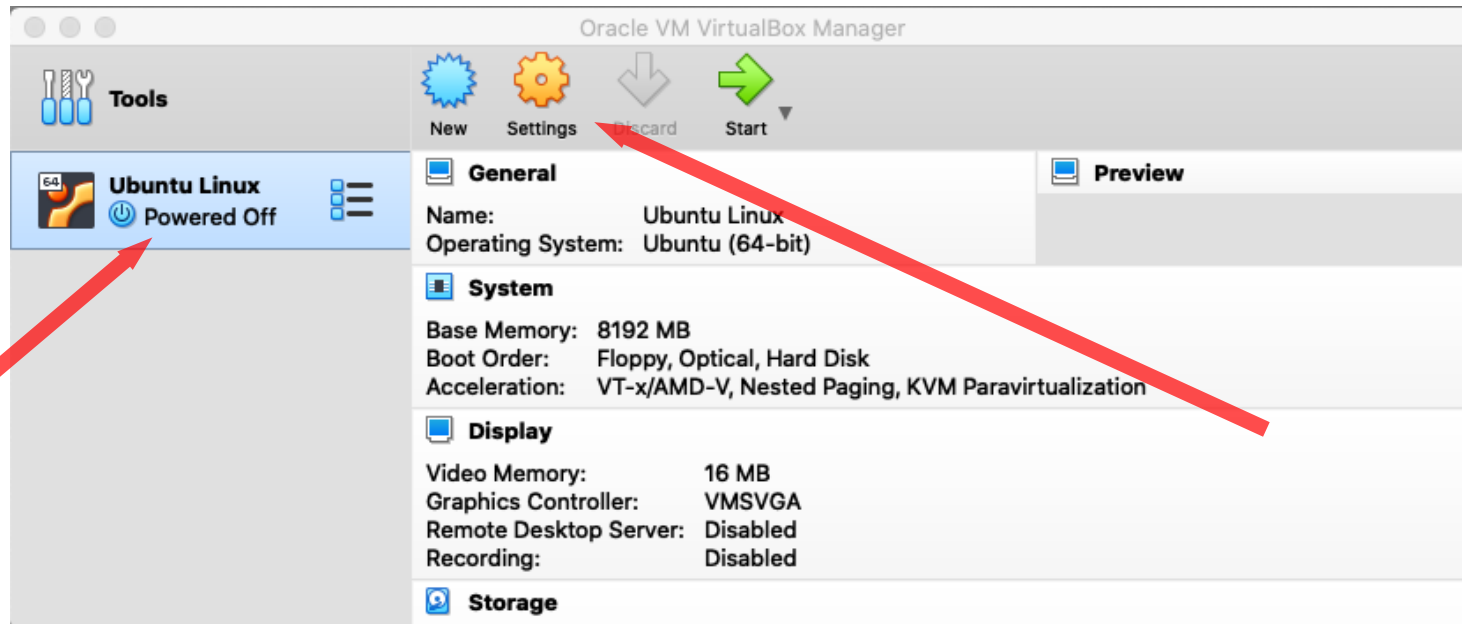
Wpisujemy dowolną nazwę a potem: Type: Linux. Version: Ubuntu (64-bit). Przydzielamy ok. połowę pamięci RAM. Klikamy „Create”.

VirtualBox



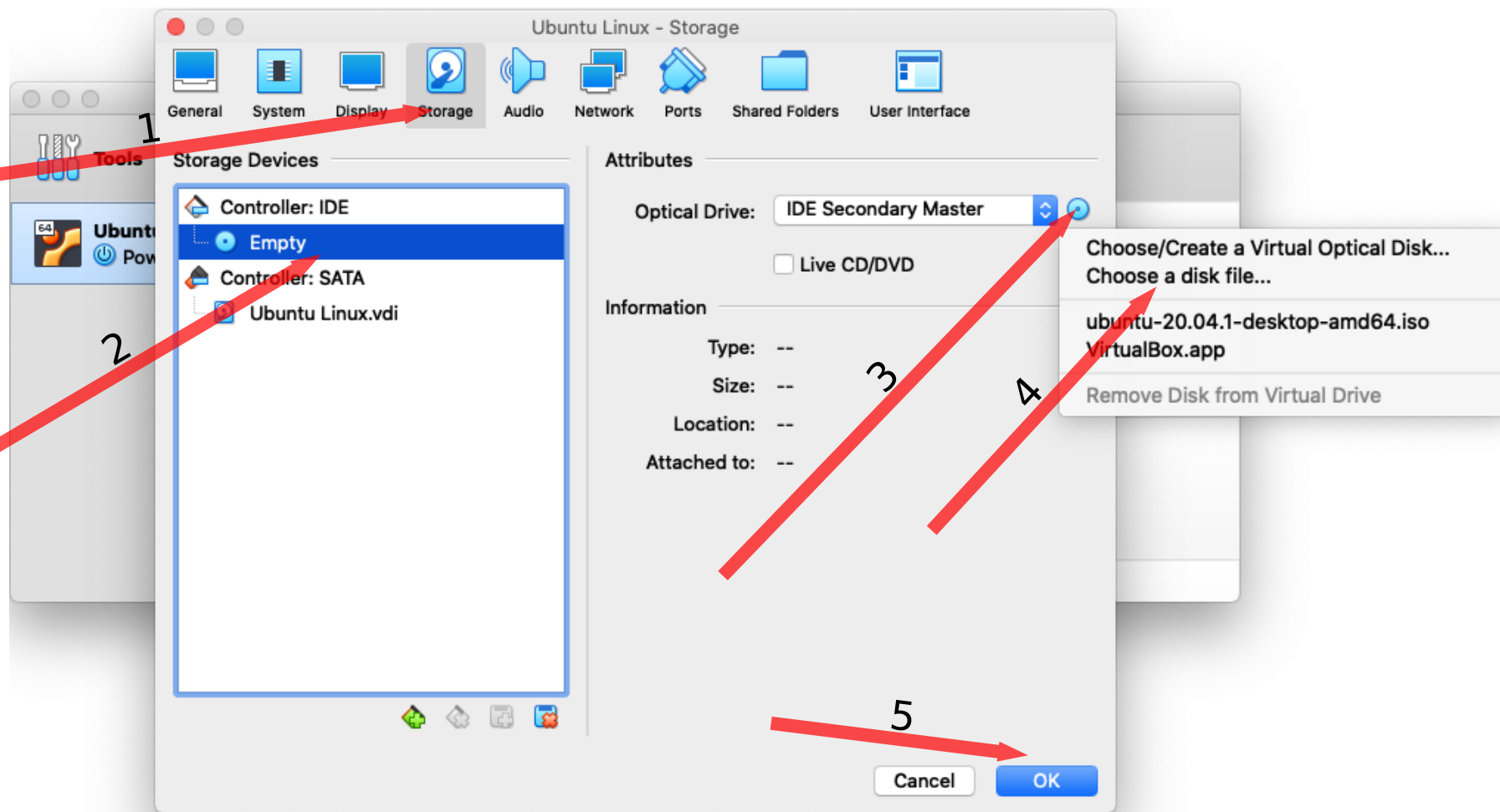
Przydzielamy ok. 64 GB dysku wirtualnemu komputerowi. Klikamy „Create”.

VirtualBox



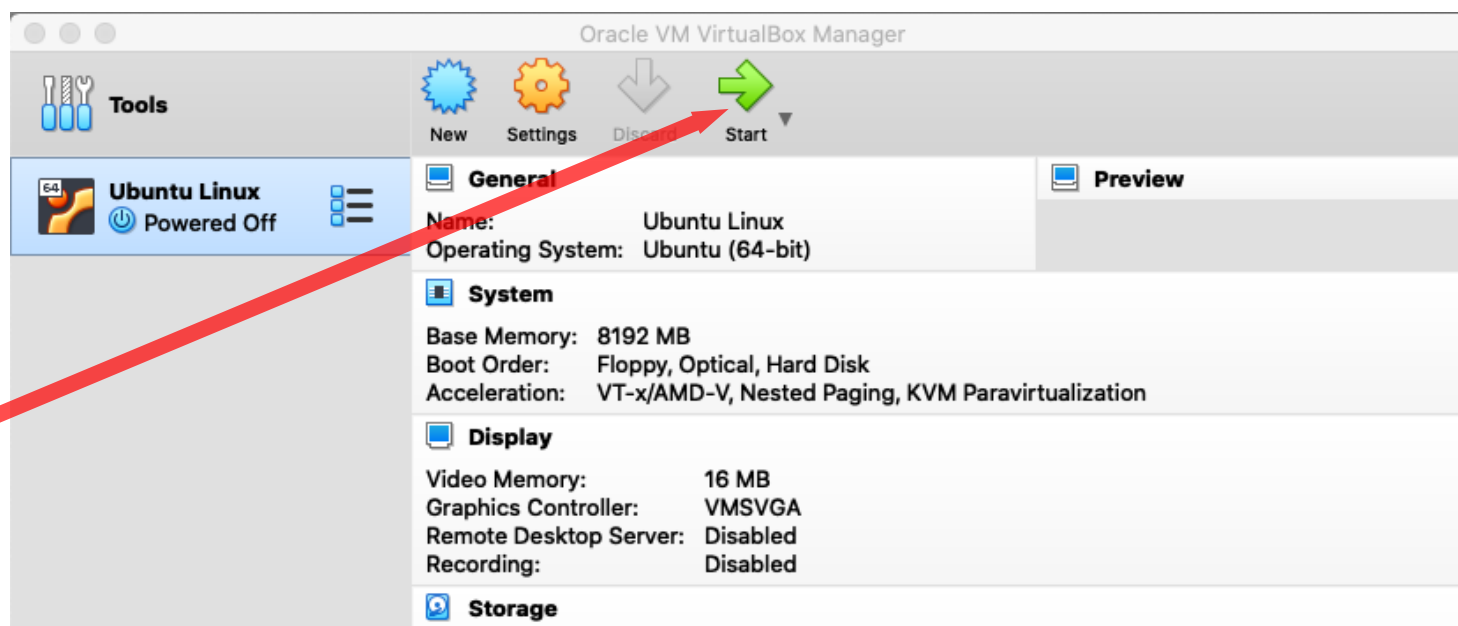
I mamy nasz nowy „wirtualny komputer na liście. Zanim go uruchomimy ściągamy Linuxa: <https://ubuntu.com/download/desktop>. Potem klikamy Settings

VirtualBox



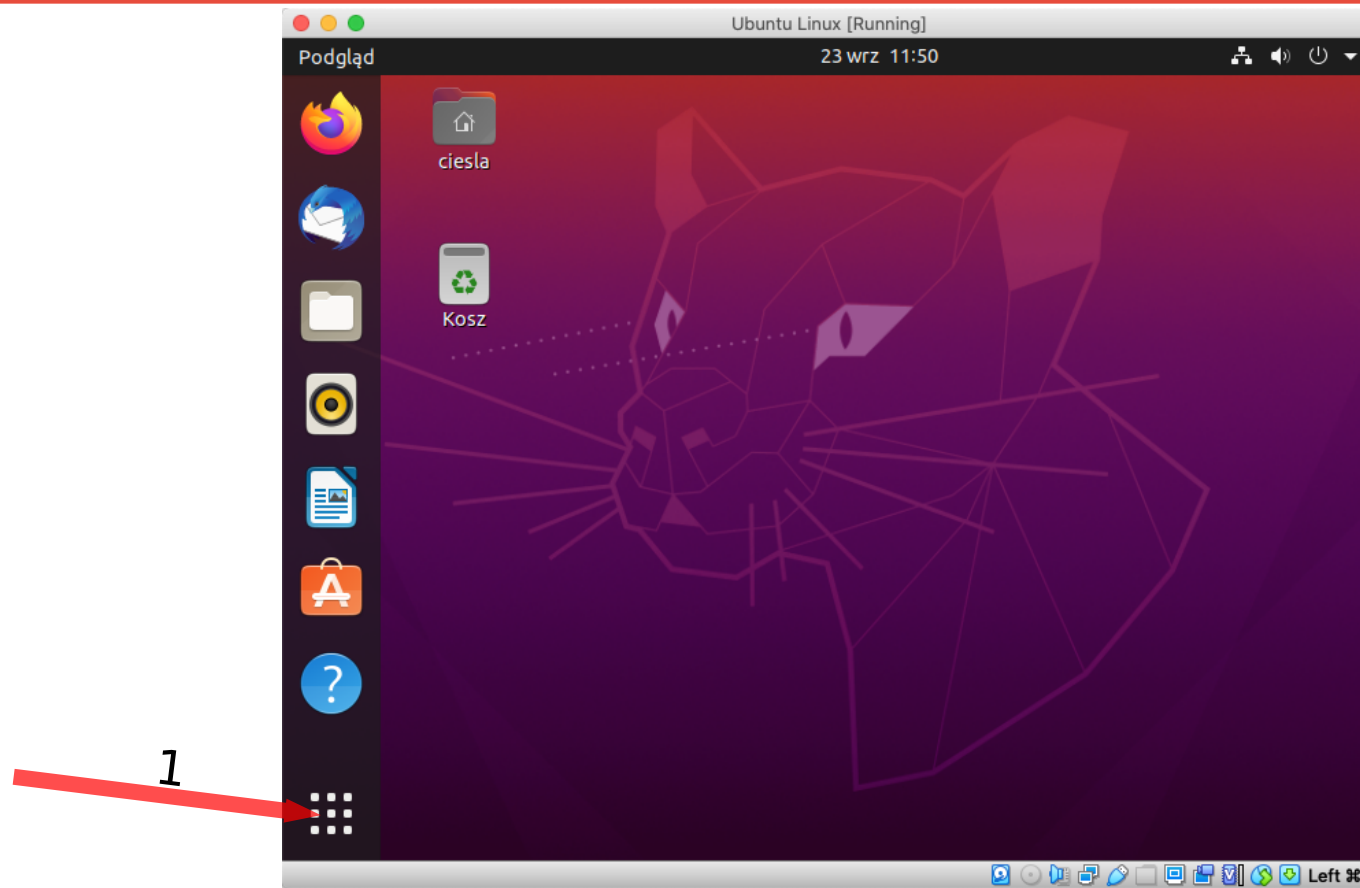
Wybieramy pobrany plik: ubuntu-....-amd64.iso. Klikamy „OK”.

VirtualBox



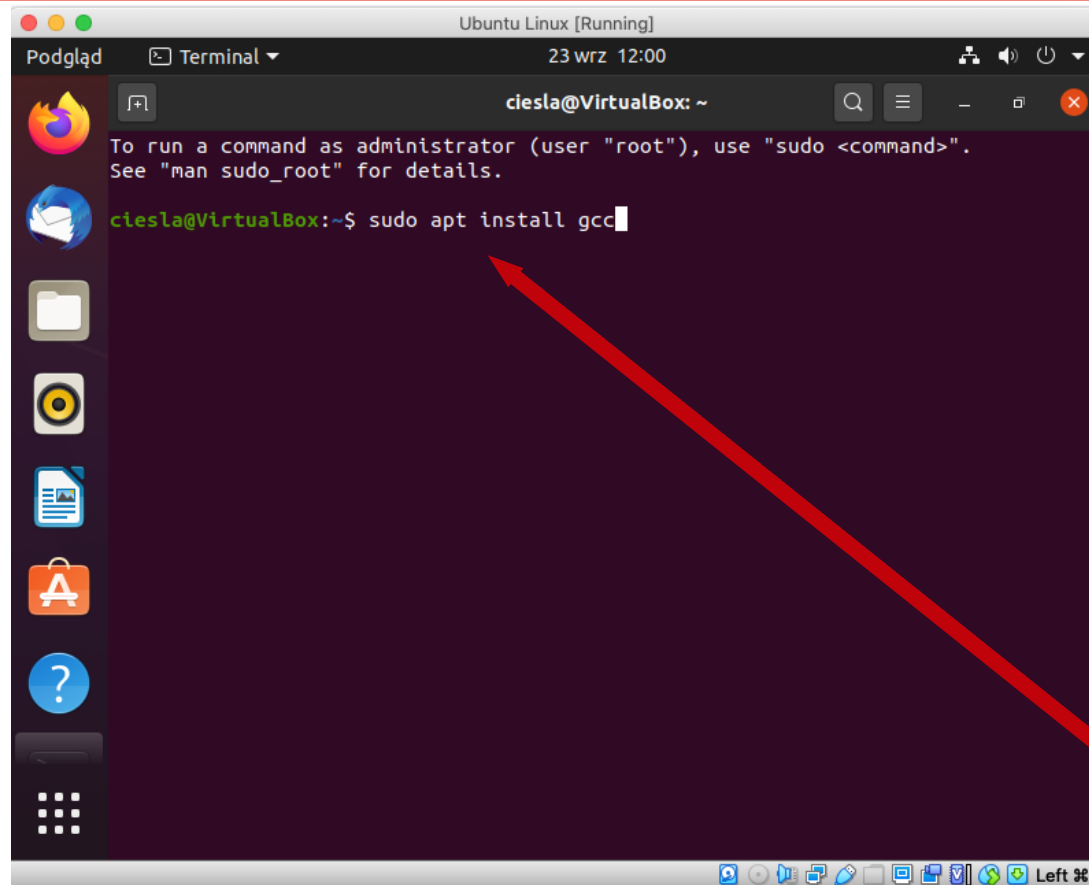
I uruchamiamy nasz „wirtualny” komputer. Po chwili rozpocznie się proces instalacji Linuxa. Instalacja trwa kilka(naście) minut. Jeśli nie wiemy co wybrać zostawiamy opcje domyślne.

Ubuntu Linux



Linux zainstalowany! Teraz instalujemy kompilator C. Klikamy w „kropki” i wyszukujemy aplikacji „Terminal”. Możemy ją sobie dodać do bocznego paska - przyda nam się jeszcze wiele razy.

Ubuntu Linux



The screenshot shows a terminal window titled "Ubuntu Linux [Running]" with a dark purple background. The prompt is "ciesla@VirtualBox: ~". The text "To run a command as administrator (user 'root'), use 'sudo <command>'. See 'man sudo_root' for details." is displayed. The command "sudo apt install gcc" is entered at the prompt. A red arrow points to the command. The terminal window has a sidebar on the left with various application icons and a top bar with system information like "Podgląd", "Terminal", and "23 wrz 12:00".

```
ciesla@VirtualBox: ~  
To run a command as administrator (user "root"), use "sudo <command>".  
See "man sudo_root" for details.  
ciesla@VirtualBox:~$ sudo apt install gcc
```

sudo apt install gcc [Enter]

Ubuntu Linux

Opcjonalnie zaleca się zainstalować VirtualBox Guest Additions. W tym celu korzystając z terminala instalujemy make i perl

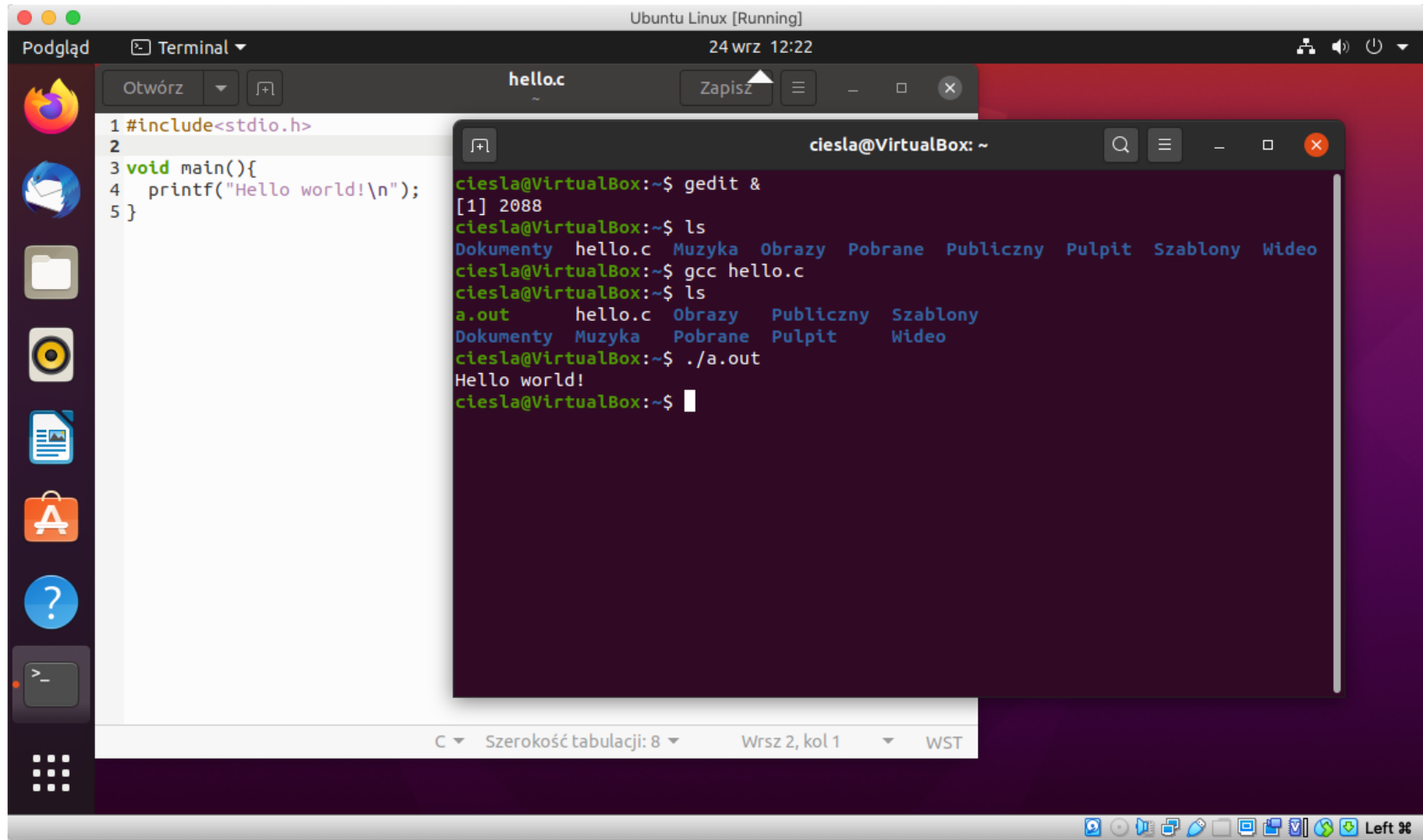
sudo apt install make perl

A następnie z menu VirtualBox wybieramy:

Devices / Insert Guest Additions CD image...

I po instalacji restartujemy Linuxa

Hello world



The image shows a screenshot of a Linux terminal window titled "Ubuntu Linux [Running]". The window has a top bar with "Podgląd", "Terminal", and "24 wrz 12:22". Below the top bar, there are window controls and a file manager view showing a file named "hello.c". The code in "hello.c" is:

```
1 #include<stdio.h>
2
3 void main(){
4     printf("Hello world!\n");
5 }
```

Overlaid on the terminal is a smaller terminal window titled "ciesla@VirtualBox: ~". This window shows the following commands and output:

```
ciesla@VirtualBox:~$ gedit &
[1] 2088
ciesla@VirtualBox:~$ ls
Dokumenty hello.c Muzyka Obrazy Pobrane Publiczny Pulpit Szablony Wideo
ciesla@VirtualBox:~$ gcc hello.c
ciesla@VirtualBox:~$ ls
a.out hello.c Obrazy Publiczny Szablony
Dokumenty Muzyka Pobrane Pulpit Wideo
ciesla@VirtualBox:~$ ./a.out
Hello world!
ciesla@VirtualBox:~$
```

At the bottom of the main terminal window, there are status indicators: "C", "Szerokość tabulacji: 8", "Wrsz 2, kol 1", and "WST". The system tray at the bottom right shows various icons and the text "Left %".

Hello world

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

```
void main() {
```

```
printf("Hello world!\n");
```

```
}
```

chcemy skorzystać z biblioteki, która zawiera m.in. funkcję `printf()`.

deklarujemy własną funkcję o nazwie **main**, która nie przyjmuje żadnych argumentów **()** i nie zwraca żadnej wartości **void**. W nawiasach klamrowych **{ }** umieszczamy definicję funkcji – czyli co ona robi.

Wywołujemy funkcję **printf**, do której przekazujemy argument **"Hello world!\n"**. Każde takie wywołanie funkcji (instrukcję) kończymy znakiem **;**.
Program zapisujemy w pliku **hello.c**.

Hello world

Przed uruchomieniem program musi zostać skompilowany - przetłumaczony na instrukcje zrozumiałe przez procesor. Kompilator to program o nazwie **gcc**:

```
gcc hello.c
```

W wyniku kompilacji powstaje plik **a.out**, który możemy uruchomić:

```
./a.out
```

Można zmienić nazwę skompilowanego pliku

```
gcc -o hello hello.c
```

a następnie

```
./hello
```

Funkcje

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

```
char* napis(){  
    return "Hello world!\n";  
}
```

```
void main(){  
    printf(napis());  
    printf("%s", napis());  
}
```

deklarujemy własną funkcję o nazwie **napis**, która zwraca wartość typu **char*** - ciąg znaków. Po słowie **return** znajduje się zwracana wartość przez funkcję

Funkcję możemy wywoływać wielokrotnie. Pierwszy sposób użycia **printf()** jest niezalecany - kompilator zwróci ostrzeżenie.

Pierwszy argument **printf()** to formatowanie **%s** mówiące, że kolejny argument ma być wyświetlony jako ciąg znaków.

Więcej informacji o formatowaniu w **printf()**:

<https://www.cypress.com/file/54441/download>

Funkcje

```
#include<stdio.h>
int dodaj(int a, int b){
    return a+b;
}

int pomnoz(int a, int b){
    return a*b;
}

void main(){
    printf("%d + %d = %d\n", 3, 5, dodaj(3,5));
    printf("%d * %d = %d\n", 3, 5,
pomnoz(3,5));
}
```

Funkcja `dodaj` ma dwa argumenty typu `int`. Jeden nazwaliśmy `a` a drugi `b`.

`%d` oznacza, że w tym miejscu zostanie wyświetlony kolejny argument funkcji `printf` w formie liczby całkowitej.

Podstawowe typy danych

int - liczba całkowita

long - liczba całkowita (większa, zapisywana z wykorzystaniem większej liczby bajtów)

short - liczba całkowita (mniejsza niż int, zapisana z wykorzystaniem mniejszej liczby bajtów)

float - liczba rzeczywista (zmiennoprzecinkowa)

double - liczba rzeczywista (większa, dokładniejsza, zapisywana z wykorzystaniem większej liczby bajtów)

char - znak (np. 'a', 'b', ' '). Znaki umieszczają się między apostrofami)

***** - ciągi, np. **char***, **int***, ...

Plan wykładu

Dziękuję za uwagę