Herzlich willkommen!

Erhebung, Analyse und Visualisierung digitaler Daten

(*Prozedurale Programmierung* | *Einführung in die Informatik*)

Prof. Dr. Sebastian Zug, Technische Universität Bergakademie Freiberg

Organisatorische Einordnung / Motivation

Das vorliegende Material bedient die bisherigen Veranstaltungen

- *Prozedurale Programmierung* und
- *Einführung in die Informatik*

und firmiert diese als neukonzeptionierte übergreifenden Vorlesung *Erhebung, Analyse und Visualisierung digitaler Daten*.

> Die Veranstaltung richtet sich an Nicht-Informatiker aus verschiedenen ingenieurwissenschaftlichen Disziplinen mit keinen oder geringen Programmierkenntnissen.

Worum geht es?

Modulhandbuch

Qualifikationsziele / Kompetenzen:

Mit der erfolgreichen Teilnahme an der Veranstaltung sollen die Studierenden:

- verstehen, was Algorithmen sind und wie konkrete wissenschaftliche Aufgaben algorithmisch abgebildet werden können,
- Konzepte der prozeduralen und objektorientiertenProgrammierung in Python und C++ anzuwenden
- in der Lage sein, praktische Herausforderungen derDatenaggregation und Verarbeitung zu identifizieren und Umsetzungen zu realisieren
- Werkzeuge der Programmierung einordnen und nutzen zu können
- Datenstrukturen und algorithmische Konzepte anwenden zu können und über Wissen ausgewählter Standardalgorithmen verfügen.

Konkrete Anwendungsszenarien

1. Erstellen einer Hypothese
2. Entwurf eines Experimentes für die systematische Untersuchung der Fragestellung
3. Durchführung des Experimentes
4. Bewertung der erlangten Daten

Beispiel

1. Hypothese: Am Wochenende scheint die Sonne häufiger als unter der Woche.

2. Konzeption eines Experimentes: ...

Frage: Wie würden Sie vorgehen?

3. Durchführung des Experimentes

Prototypische Wetterstation im Keller des Humboldt-Baus

4. Filterung und Interpretation der erlangten Daten

Das Diagramm zeigt die Darstellung der Lichtintensität über den Stunden eines Tages für eine Woche. Blau sind die Wochentage markiert, rot der Samstag und der Sonntag.

Frage: Welche Kritik sehen Sie an dieser Methodik?

Weitere Beispiele:

- Erfassen und Analysieren Sie das Insektenaufkommen auf dem Campus der TU Freiberg!
- In welchem Zusammenhang steht der Energieverbrauch der Gebäude der Bergakademie mit der Außentemperatur?
- An welchen Stellen in Freiberg muss unser Roboter häufiger stoppen, um Passantinnen queren zu lassen?

Merke: Wissenschaftliches Arbeiten ist im Bereich der Natur- und Ingenieurwissenschaften ohne den Rechner (fast) nicht denkbar.

Unsere Motivation

- **Anwendungssicht**

  Wir möchten Sie in die Lage versetzen einfache Messaufgaben (mit einem Mikrocontroller) zu realisieren und die Daten auszuwerten.

- **Algorithmische Perspektive**

  Wir möchten Sie dazu ertüchtigen den Algorithmusbegriff der Informatik zu durchdringen und anwenden zu können.

- **Konzeptionelle Perspektive**

  Sie erlernen grundlegende Elemente der prozeduralen und der objektorientierten Programmierung.

- **Umsetzungssicht**

  Wir vermitteln Grundkenntnisse in den Programmiersprachen C++ und Python.

Zwischenfrage: Welche Argumente vermuten Sie hinter der Entscheidung zwei Programmiersprachen in die Vorlesung aufzunehmen?

	Phase 1	**Phase 2**
Programmiersprache	C++	Python
Framework / Packages	Arduino	pandas/numpy/matplotlib
Ziel	Datenerhebung	Datenauswertung
Plattform	PC / Mikrocontroller	PC

Lernziele der Vorlesung

Einordnung und Klassifikation	**Studierende ...**
Erschaffen	-
Bewerten	• ... können die Unterschiede der behandelten Programmiersprachen mit Bezug auf Ausführungskontex, Performance usw. beurteilen .
Analysieren	• ... bewerten aggregierte Messdaten mit Blick auf deren Aussagekraft mit statistischen Methoden. • ... können die Eigenschaften eines Sensors anhand von Datenblättern recherchieren. • ... quantifizieren das Zeitverhalten in Datensätzen und erklären es anhand des Programmcodes für die Datenerhebung.
Anwenden	• ... wenden die Basis-Techniken der Codeentwicklung, des Debuggings und der Dokumentation an. • ... sind in der Lage im Arduino-Kontext nach geeigneten Open-Source-Paketen zu suchen. • ... realisieren kleiner Mikrocontroller-Beispiele auf Basis des Arduino-Projektes bzw. einer Datenanalysepipeline mit Python.
Verstehen	• ... sind in der Lage den Algorithmusbegriff zu erklären. • ... können die Elemente prozeduraler Programmierung (Schleife, Verzweigung, Sprung) beschreiben. • ... erklären Basiskonzepte objektorientierter Programmierung (Vererbung, Kapselung).
Erinnern	• ... kennen die grundlegende Syntaxelemente der behandelten Programmiersprachen.

Keine Angst vor Code!

Das Beispiel zeigt ein "Hello World" Beispiel für die C++ Implementierung auf dem Mikrocontroller. Um das einfachste Beispiel darzustellen, lesen wir keinen Sensor ein, sondern lassen eine kleine LED blinken.

Hello World - Beispiele Phase 1

```cpp
#include <iostream>
using namespace std;

int main (){
  int i = 0;
  int max = 0;

  cout << "How many hellos: ";
  cin >> max;

  for(i=0; i<max; i++)
    cout << "Hello, world " << i << endl;

  return 0;
}
```

```
How many hellos:
```

Auf der Basis von C++ lassen sich einfache Mess- und Regelungstechnikaufgaben sehr einfach automatisieren. Ein Startpunkt dafür ist das Arduino Projekt, dass sowohl eine entsprechende Hardware, wie auch eine Programmierumgebung bereitstellt.

helloWorldinArduino.cpp

```cpp
void setup() {
  pinMode(LED_BUILTIN, OUTPUT);
}

void loop() {
  digitalWrite(LED_BUILTIN, HIGH);
  delay(1000);
  digitalWrite(LED_BUILTIN, LOW);
  delay(1000);
}
```

```
Sketch uses 924 bytes (2%) of program storage space. Maximum is 32256
bytes.
Global variables use 9 bytes (0%) of dynamic memory, leaving 2039 bytes
for local variables. Maximum is 2048 bytes.
```

Die Programmiersprache Python offeriert eine Vielzahl von Paketen für unterschiedlichste Zwecke. Wir werden uns auf die Visualisierung und Datenauswertung konzentrieren.

Hello World - Beispiele Phase 2

data.csv

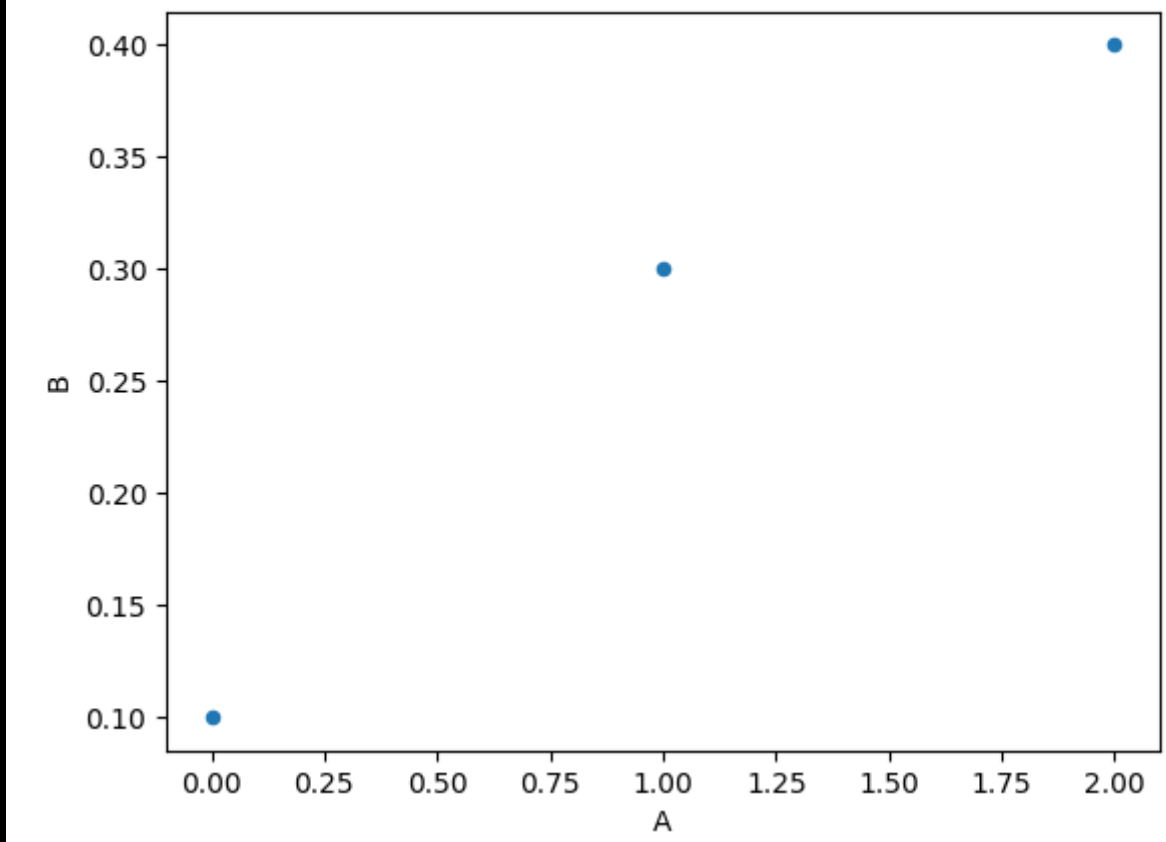
```
A,B,C
0,0.1,3
1,0.3,5
2,0.4,2
```

readCSV.py

```
import pandas as pd
import matplotlib.pyplot as plt

df = pd.read_csv('data.csv', header = 0)
df.plot.scatter(x='A', y='B')
plt.savefig('temp.png')
```

temp.png



temp.png data.csv main.py

"Python ist ja nett, aber C++ ist ..."

Foreneinträge aus Veranstaltungen anderer Hochschulen:

> *"Viele haben bei uns wegen dem Info-Grundlagenmodul gewechselt. Allerdings hängt das auch von dir und deinem Talent ab. Das Tempo ist rasant. Jede Art von Vorerfahrung hilft dir eigentlich sehr. Also wenn du noch Zeit hast vorm Studienbeginn, schnapp dir ein gutes Buch zur gelehrten Sprache, und fange schonmal bissl an kleine Sachen zu programmieren."*
>
> *"Ich habe es gerade irgendwie selbst gelöst, aber keine Ahnung warum es funktioniert hat."*

Frage: Was sind die besonderen Herausforderungen bei der Programmierarbeit?

- Abstrakte Denkweise
- Penible Beachtung der Syntax
- Ungewohnte Arbeitsmittel

BuggyCode.cpp

```
 1 void setup() {
 2   pinMode(LED_BUILTIN, OUTPUT)
 3 }
 4
 5 void loop() {
 6   digitalWrite(LED_BUILTIN, HIGH);
 7   delay(1000);
 8   digitalwrite(LED_BUILTIN, LOW);
 9   delay(1000.);
10
```

```
/sketch/sketch.ino: In function 'void setup()':
/sketch/sketch.ino:3:1: error: expected ';' before '}' token
 }
 ^
/sketch/sketch.ino: In function 'void loop()':
/sketch/sketch.ino:8:3: error: 'digitalwrite' was not declared in this
scope
   digitalwrite(LED_BUILTIN, LOW);
   ^~~~~~~~~~~~
/sketch/sketch.ino:8:3: note: suggested alternative: 'digitalWrite'
   digitalwrite(LED_BUILTIN, LOW);
   ^~~~~~~~~~~~
   digitalWrite
/sketch/sketch.ino:9:15: error: expected '}' at end of input
   delay(1000.);
               ^

Error during build: exit status 1
```

Wo stehen Sie?

Wer von Ihnen hat bereits Programmiererfahrung? Was würden Sie auf die Frage antworten, wie viele Codezeilen Sie bereits geschrieben haben?

- ◯ Keine einzige Zeile
- ◯ 20 Zeilen in der Schule
- ◯ n Zeilen in eigenen Programmierprojekten
- ◯ sehr viele Zeilen in Open Source Projekten

Organisatorisches

Wer sind *wir*?

Name	Email	Rolle
Prof. Dr. Sebastian Zug	sebastian.zug@informatik.tu-freiberg.de	Vorlesungen
Dr. Galina Rudolf	galina.rudolf@informatik.tu-freiberg.de	Vorlesungen und Koordination der Übungen
Florian Richter	flo.richter@informatik.tu-freiberg.de	
Robert Lösch	robert.loesch@informatik.tu-freiberg.de	
Adrian Köppen	adrian.koeppen@student.tu-freiberg.de	Tutor

Strukurierung der Veranstaltung

Nr.	Datum		Inhalt
0	16.10.2023		Motivation, Organisation
1	23.10.2023	C++	C++ Programmstrukturen / Entwicklungsprozess
2	30.10.2023	C++	Datentypen / Ein- und Ausgabe
3	06.11.2023	C++	Kontrollstrukturen
4	13.11.2023	C++	Zeiger und Arrays
5	20.11.2023	C++	Funktionen
6	27.11.2023	C++	Objekte
7	04.12.2023	C++	Vererbung
8	11.12.2023	C++	Anwendung auf dem Mikrocontroller
9	18.12.2023	Python	Python Grundlagen
10	08.01.2023	Python	Python Grundlagen
11	15.01.2023	Python	Objekte
12	22.01.2023	Python	Visualisierung
13	29.01.2023	Python	Datenanalyse
14	05.02.2023	Python	Übergreifende Anwendungen

Achtung: C++ wird zunächst ohne die spezifische Verwendung des Mikrocontrollers verwendet. Vielmehr erfolgt die Programmierung auf dem Desktoprechner. Wenn wir die Grundlagen erarbeitet haben, wechseln wir die Hardware einfach aus.

Übungen

Die Übungen vertiefen das erlernte anhand praktischer Programmieraufgaben:

... haben wir hier noch mal ein Beispiel?

> **Frage:** Ist jemand noch nicht für die Übungen eingeschrieben?

> **Empfehlung:** Für die Arbeit am Programmcode empfehlen wir den freien Editor Visual Studio Code. Dieser kann sowohl für die Arbeit mit den C++ als auch Python Programmbeispielen genutzt werden.

Mikrocontroller Tutorials

Die Tutorials werden im Dezember erstmalig angeboten. Sie dienen interessierten Studierenden der weiteren Vertiefung ihrer Kenntnisse.

> Mikrocontrollerbezogene Inhalte sind nicht Gegenstand der Prüfungen.

Vorlesungsmaterialien

Die Vorlesungsunterlagen selbst sind unter

https://github.com/SebastianZug/VL_EAVD

verfügbar. Diese können entweder in der Markdown-Syntax oder als interaktives Dokument betrachtet werden.

> Achtung! Es handelt sich um "lebende" Materialien.
>
> - der Inhalt wird sich ggf. anhand Ihrer Verbesserungsvorschläge verändern
> - die Dokumente enthalten eine Vielzahl von ausführbarem Code.

Erwartungen

Der Zeitaufwand beträgt 180h und setzt sich zusammen aus 60h Präsenzzeit und 120h Selbststudium. Letzteres umfasst die Vor- und Nachbereitung der Lehrveranstaltungen, die eigenständige Lösung von Übungsaufgaben sowie die Prüfungsvorbereitung.

Sie müssen, insbesondere wenn Sie noch keine Programmiererfahrung haben, ggf. deutlich mehr Zeit in den Kurs investieren. Dies macht gemeinsam mehr Spaß als allein!

Prüfungsinhalte

Die Prüfung besteht aus einer Klausur und ggf. einer praktischen Arbeit (Einführung in die Informatik).

- *Entwickeln Sie ein Programm, das ...!*
- *Beschreiben Sie die Funktionsweise des folgenden Programms ...!*
- *Welchen Wert gibt das folgende Programm in Zeile x aus?*
- *Finden Sie alle syntaktischen und logischen Fehler im nachfolgenden Code*
- *Bewerten Sie folgenden Aussagen: ...*

Wie können Sie zum Gelingen der Veranstaltung beitragen?

- Stellen Sie Fragen, seinen Sie kommunikativ!

> Hinweis auf OPAL Forum!

- Organisieren Sie sich in Arbeitsgruppen!

> Hinweis auf Repl.it

- Machen Sie Verbesserungsvorschläge für die Vorlesungsfolien!
- Sprechen Sie uns gern wegen "Bastelbedarf" für ein eigenes Projekt an!

Beispiel der Woche

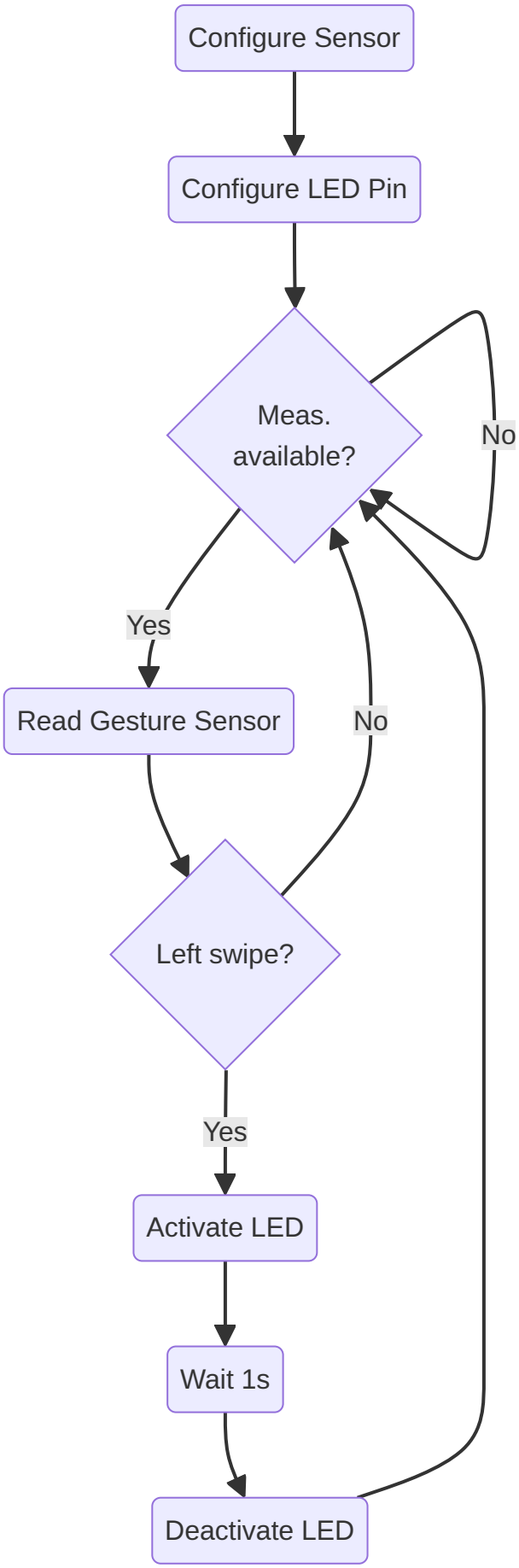
Aufgabe: Wir Gestenbasiertes Schreibtischlicht realisieren - Auf eine Bewegung der Hand nach links hin soll sich das Licht einschalten und nach einer gewissen Zeit (1s) wieder ausschalten.

Grundlage für die Umsetzung ist ein Microcontroller mit einem APDS-9960 Sensor. Dieser Sensor kann einfache Gesten erkennen.

> Welche "Bausteine" brauchen wir dafür?

> Wie müssen diese miteinander verknüpft werden?

Wie kann ich diese Information abstrakt darstellen?

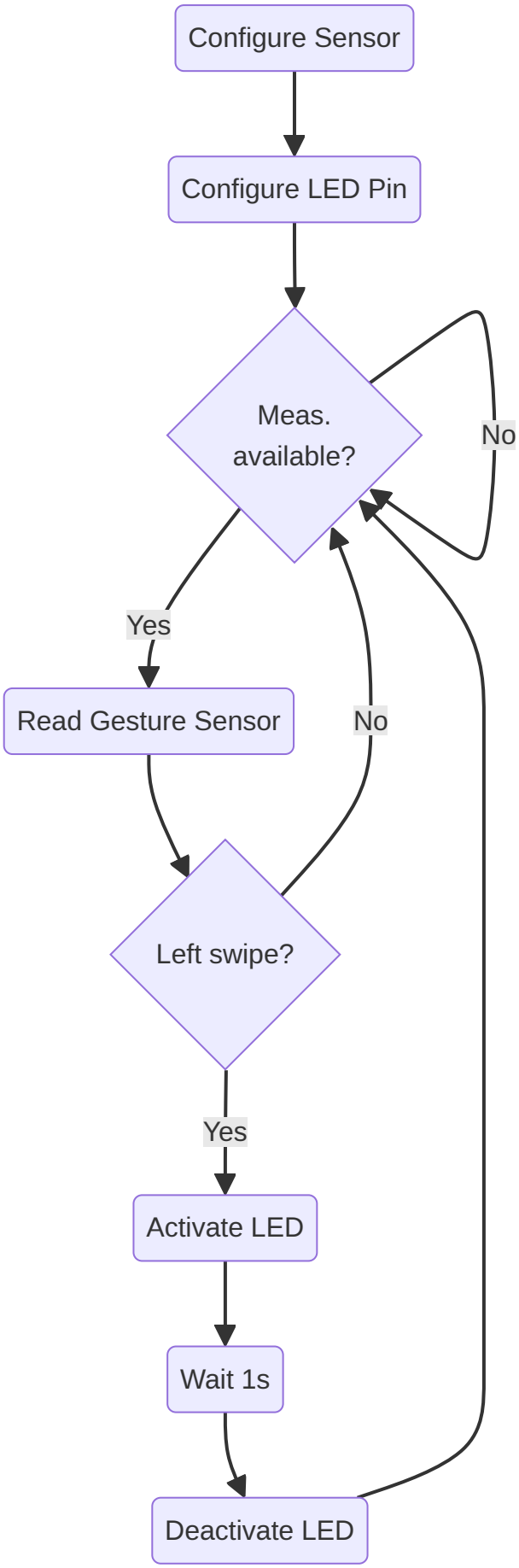

Configure Sensor
Configure LED Pin
Meas. available?
No
Yes
Read Gesture Sensor
No
Left swipe?
Yes
Activate LED
Wait 1s
Deactivate LED

Wie sieht die C++ Lösung dafür aus?

```
#include <Arduino_APDS9960.h>

void setup() {
  APDS.begin();
  pinMode(LED_BUILTIN, OUTPUT);
}

void loop() {
  if (APDS.gestureAvailable()) {
    int gesture = APDS.readGesture();
    if (gesture == GESTURE_LEFT)
    {
        digitalWrite(LED_BUILTIN, HIGH);
        delay(1000);
        digitalWrite(LED_BUILTIN, LOW);
    }
  }
}
```


Configure Sensor
Configure LED Pin
Meas. available?
No
Yes
Read Gesture Sensor
No
Left swipe?
Yes
Activate LED
Wait 1s
Deactivate LED