Studenta īsā rokasgrāmata

Programmēšanas un datorizētu sistēmu pamati - lietošanai nodarbībā un pašmācībā

Kursa pamats: Programmēšanas un datorizētu sistēmu pamati v2.0 FINAL

Darba mērķis: ātra lietošana nodarbībā, pašmācībā un projekta izstrādē.

Kā lietot šo rokasgrāmatu

Šī ir īsā darba versija studentam. Tā neaizstāj pilno 560 lappušu kursa materiālu, bet palīdz ātri atrast būtiskāko: principus, komandas, formulas, laboratoriju plūsmu un projekta dokumentācijas secību.

Marķējums	Nozīme	Ko darīt
PAMATI	Obligātais koledžas līmenis	Jāsaprot un jāspēj izpildīt bez palīdzības.
PROFESIONĀLI	Saikne ar reālu inženiertehnisku darbu	Jāspēj pielietot laboratorijā vai projektā.
UNIVERSITĀTE	Padziļinājums robotikai/mehatronikai	Jāspēj paskaidrot, ja izvēlas sarežģītu projektu.

Kursa ātrā karte

Modulis	Ko iemācies
1. Algoritmi	Problēma -> soļi -> blokshēma -> pseidokods -> tests
2. Python	Mainīgie, ievade/izvade, if/elif/else, cikli, funkcijas, faili, datu apstrāde
3. C++	Kompilācija, datu tipi, struktūra, cikli, funkcijas, masīvi, klases, state machine
4. SQL un IoT	Dati tabulās, INSERT/SELECT, CSV/JSON, MQTT/HTTP, edge/cloud
5. Mehatronika	Sensori, aktuatori, feedback, PID, interlock, drošības loģika
6. Robotika	Kinemātika, LiDAR, IMU, Machine Vision, SLAM, ROS 2, autonomā navigācija
7. Projekts	Functional Tree -> prasības -> kods -> tests -> pierādījumi -> aizstāvēšana

Algoritmu minimums

- Algoritms ir galīga, precīza darbību secība.
- Katram algoritmam jābūt saprotamam, izpildāmam, galīgam un pārbaudāmam.
- Vienmēr pieraksti ievades, apstrādi un izvades.
- Pirms koda jābūt algoritmam vai Functional Tree.

Jēdziens	Nozīme	Piemērs
Secība	Darbības notiek noteiktā kārtībā	Nolasīt sensoru -> aprēķināt -> izvadīt rezultātu
Zarošanās	Lēmums pēc nosacījuma	Ja T > 30, ieslēgt ventilatoru
Cikls	Atkārtošana līdz mērķim vai nosacījumam	Kamēr nav sasniegts 3 m, turpināt braukt

Python quick reference

- Python lieto ātrai prototipēšanai, datu apstrādei, testiem un robotikas simulācijām.
- Svarīgāk ir saprast datu plūsmu, nevis iekalt komandas.

Jēdziens	Nozīme	Piemērs
Ievade	input()	name = input("Name: ")
Izvade	print()	print("OK")
Nosacījums	if / elif / else	if t > 30:
Cikls	for / while	for i in range(10):
Funkcija	def	def read_sensor():
Saraksts	list	values = [1,2,3]
Faili	open()	with open("data.csv") as f:

C++ quick reference

- C++ palīdz saprast tipu, kompilācijas un aparatūrai tuvākas programmas.
- Tehniskajos projektos C++ bieži parādās Arduino/ESP32/robotu vadībā.

Jēdziens	Nozīme	Piemērs
Izvade	cout	cout << "OK";
Ievade	cin	cin >> value;
Tips	int/float/bool/string	float temp = 24.5;
Nosacījums	if/else	if (temp > 30) { }
Cikls	for/while	for (int i=0;i<10;i++)
Funkcija	return type	float average(float a, float b)
Klase	class	class Sensor { };

SQL un datu pamati

- SQL izmanto, lai strukturēti glabātu un atlasītu datus.
- IoT projektā dati nav tikai ekrāna teksts - tie jāglabā, jāfiltrē un jāanalizē.

Jēdziens	Nozīme	Piemērs
Tabula	Datu struktūra	measurements(id, temperature, timestamp)
INSERT	Pievieno ierakstu	INSERT INTO measurements ...
SELECT	Atlasa datus	SELECT * FROM measurements
WHERE	Filtrs	WHERE temperature > 30
ORDER BY	Kārtošana	ORDER BY timestamp DESC
JOIN	Tabulu savienošana	devices JOIN measurements

IoT un mākoņdati

- IoT ķēde: sensors -> ierīce -> tīkls -> brokeris/API -> datubāze -> vizualizācija.
- Vienmēr domā par laika zīmogu, savienojuma zudumu un datu drošību.

Jēdziens	Nozīme	Piemērs
MQTT	Publisher/subscriber datu apmaiņa	topic: lab/room1/temp
HTTP/REST	Pieprasījums/atbilde	GET /api/measurements
JSON	Datu formāts	{"temp":24.5}
Edge	Apstrāde pie ierīces	filtrs pirms sūtīšanas
Cloud	Datu glabāšana/analīze	dashboard + SQL

Mehatronikas minimums

- Mehatronika apvieno mehāniku, elektroniku, vadību un programmatūru.
- Vadības cilpa vienmēr jāskata kopā ar fizisko sistēmu un drošību.

Jēdziens	Nozīme	Piemērs
Sensors	Mēra procesu	temperatūra, attālums, IMU
Aktuators	Izdara fizisku darbību	motors, servo, relejs
Feedback	Atgriezeniskā saite	salīdzina mērķi ar mērījumu
PID	Regulators	P + I + D reakcija uz kļūdu
Interlock	Drošības nosacījums	durvis aizvērtas AND E-stop nav aktīvs

Robotikas minimums

- Autonoms robots darbojas ciklā: Sense -> Perceive -> Localize/Map -> Decide -> Plan -> Control -> Act.
- Pirms autonomijas jāpārbauda katra datu plūsma atsevišķi.

Jēdziens	Nozīme	Piemērs
LiDAR	Attāluma skenēšana	/scan
IMU	Paātrinājums un rotācija	/imu/data
Odometrija	Robota kustības novērtējums	/odom
SLAM	Karte + lokalizācija	map + pose
Nav2	Autonomā navigācija	NavigateToPose
TF	Koordinātu transformācijas	map -> odom -> base_link

Projekta pierādījumu ķēde

- Labs projekts nav tikai kods. Tam jābūt pierādāmam.
- Katram apgalvojumam jābūt sasaistītam ar testu vai mērījumu.

Jēdziens	Nozīme	Piemērs
1	Problēmas apraksts	Ko sistēma risina?
2	Functional Tree	Kā sistēma sadalās funkcijās?
3	Prasības	Ko tieši sistēmai jādara?
4	Kods	Kā prasība realizēta?
5	Tests	Kā pierādīts, ka strādā?
6	Aizstāvēšana	Ko students spēj paskaidrot?

Debugging lēmumu ceļvedis

Simptoms	Pārbaudi vispirms	Nākamais solis
Programma nepalaižas	Sintakse, importi, kompilācijas kļūdas	Samazini piemēru līdz minimālam kodam.
Rezultāts nepareizs	Ievades un starprezultāti	Izdrukā mainīgos pa soļiem.
Sensors rāda muļķības	Barošana, GND, adrese, pieslēgums	Pārbaudi sensoru ar atsevišķu testprogrammu.
Motors nestrādā	Atsevišķa motora barošana, draiveris, GND	Atdali loģiku no jaudas ķēdes.
ROS 2 dati nav redzami	topic nosaukums, QoS, source setup	ros2 topic list/echo/hz, rqt_graph.

Pašvērtējuma lapa

Prasme	Vērtējums 0-3	Pierādījums
Izveidoju algoritmu pirms koda		
Varu paskaidrot Python programmu		
Varu paskaidrot C++ programmu		
Protu pierakstīt SQL vaicājumu		
Saprotu sensoru datu plūsmu		
Varu izskaidrot autonomā robota ciklu		
Protu testēt un dokumentēt rezultātu		