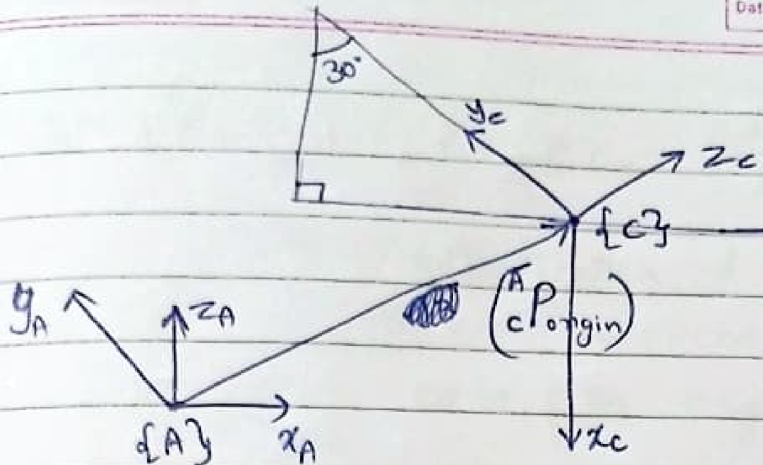
Q.1)



→ Without interference of {B}, we can directly correlate or map the two frames {A}, {C}.

∴ As we know, by convention,

$\boxed{{}^{A}_{C}T}$ = Transformation matrix for ~~mapping~~ transformation (translat^n + rotation) of frame (A) to frame (C).

$$= \left[\begin{array}{ccc|c} & {}^{A}_{C}R & & {}^{A}_{C}P_{origin} \\ \hline 0 & 0 & 0 & 1 \end{array}\right]$$

where,

- ${}^{A}_{C}P_{origin}$ = position vector of $\{C\}_{origin}$ wrt {A}. (3×1 matrix)
- {A} = frame-A.
- ${}^{A}_{C}R$ = Rotation matrix (3×3), mapping rotated frame {C} wrt {A}.

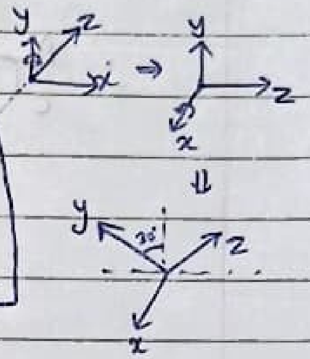
$$\therefore \; {}^{A}_{C}P_{origin} = \begin{bmatrix} 3 \\ 0 \\ 2 \end{bmatrix}_{3\times1}.$$

$\therefore \; {}^{A}_{C}R = R_x \cdot R_y \cdot R_z$

= Multiplication of individual rotation matrices about each axis, in any order.

Thus, in this case,

we can observe,



i.e. (-90°) rotation about y-axis
then,
$(+30^\circ)$ rotation about x-axis

$\Downarrow$

$\theta_x = (+30^\circ)$

$\theta_y = (-90^\circ)$

$\theta_z = (0^\circ)$

let counter-clockwise = (+ve)

$\therefore$ As we know, (using vector components and analysis)

$$R_x = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta_x & -\sin\theta_x \\ 0 & \sin\theta_x & \cos\theta_x \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \sqrt{3}/2 & -1/2 \\ 0 & 1/2 & \sqrt{3}/2 \end{bmatrix}$$

$$R_y = \begin{bmatrix} \cos\theta_y & 0 & \sin\theta_y \\ 0 & 1 & 0 \\ -\sin\theta_y & 0 & \cos\theta_y \end{bmatrix} = \begin{bmatrix} 0 & 0 & -1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix}$$

$$R_z = \begin{bmatrix} \cos\theta_z & -\sin\theta_z & 0 \\ \sin\theta_z & \cos\theta_z & 0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} = \text{(I)}$$

$$\therefore\ {}^{A}_{C}R = R_x \cdot R_y \cdot R_z = R_x \cdot R_y \cdot I$$

$$= R_x \cdot R_y$$

$$= \begin{bmatrix} 0 & 0 & -1 \\ -1/2 & \sqrt{3}/2 & 0 \\ \sqrt{3}/2 & 1/2 & 0 \end{bmatrix}$$

$$\therefore\ {}^{A}_{C}T = \begin{bmatrix} 0 & 0 & -1 & 3 \\ -1/2 & \sqrt{3}/2 & 0 & 0 \\ \sqrt{3}/2 & 1/2 & 0 & 2 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Q.2) → • As our current goal of the upcoming Humanoid robot is to walking through the Lecture halls, Lib. and campus and helping out students and faculties in various daily tasks by interacting with users (humans) and mimicing human tasks as well as enhancing work effeciencies of humans by robotically accurate works/tasks.

- We also aim to make it as social & user-friendly as possible, thus trying to mimic human behaviour.
- Thus all these aims/goals requires the robot to be accurate, efficient, and properly managed/controlled.
- Also, in doing these tasks and also in robotic competitions, it many a times requires pin-point movements & rotations & D.o.F., which will only be obtained by proper control & coordination bet^n hardware, software & sensors in the robot.

- qualitatively → Thus in controlling the humanoid, we need the skills of the control subsystem, as well as
  - proper combinat^n/ coordinat^n of other tech. subsystems
- Quantitively,

  Controlling a humanoid requires :-

  I) → sensors (Ex:- gyroscopes, touch sensors, force/torque measuring sensors, cameras, LiDAR, IMU, MPU etc.)

→ sensor are basically required to have knowledge of the surrounding of the robot & its effect on the robot.

2) → [Motors, Joints & Actuators] → ex: BO motors, DC motors, AC motors, stepper motors, servo motors, encoders, etc.

3) → Proper circuit managment & maintaining proper (sufficient & req.) current and voltage supplies.

4) → Synchronisation of motors, joints, actuators, sensors together to do a particular task accurately/precisely.

5) → Coding and Localization techniques.
(ex: SLAM technique)
(Algorithms which automate some necessary tasks, fused together with manual controlable robot.)

ex: If we manually have just told/signalled the robot to move ahead, it should move ahead but while checking for safe surrounding and potential collisions / imbalance, this makes the robot into a • humanoid.

6) AI & ML (broadly).

7) Proper communication interface betn robot & user.

→ The robot requires autonomous controls when/in the following situations :-

→ Disasteric / Unintentional unwanted/ imbalance/ collision situation.

→ In the above situation, it should automatically get back to a safe state.

→ An autonomous robot means a robot which decides its course of action by itself when a situation arises, even if the situation is not pre-programmed.
∴ Which includes the intervention of autonomous learning, deep learning, AI. etc. to be closely associated with its sensors and control interface.

→ Also, acc. to torque measuring sensors and gyroscope, the humanoid should constantly keep itself balanced on any type of terrain.

→ It should automatically periodically send maintanence feedback to the owner.

→ In case of power / batteries ~~getting~~ drained, (soon about to go), the humanoid should warn the human-controller and should reach out to a pre-programmed nearest station of charging / help desk.

Q. 3) (a) As we know the Grübler's formula, (in which we assume all constraints to be independent)

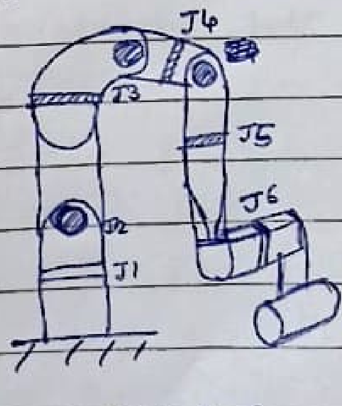
→ Here, in both these robotic arms, all the joints are independent of each other, because any of the particular joint can be moved while keeping the other joints fixed.

→ $$D.O.F. = m(N-1-J) + \sum_{i=1}^{J} (f_i)$$

where,
- N = no. of bodies (incl. ground)
- J = no. of joints
- m = DoF of a single body [6 = for spatial body (moving in 3D); 3 = for planar motion]

(f_i = no. of freedoms allowed by joint (i))

$$\therefore DOF_1 = 6(7-1-6) + 6 = 6 \text{ DoF}$$



$m = 6$ (moves in 3D space)

$\boxed{f_i = 1}$ ($\forall i$)

$J = 6$

$N = 7$

$$DOF_2 = 6(7-1-6)+6 = 6 \text{ DOF}$$

$m = 6$ (moves in 3D space)
$N = 7$
$J = 6$ (6 A's are shown in the diagram, representing 7 joints/actuators)

⇒ They have equal no. of DOF, but the arm's flexibility & range of motion also depends on <u>size and shape of bodies (components)</u> and <u>type of joints</u> & <u>their positioning</u>, i.e. the <u>dependency of constraints.</u>

⇒ In general, more the DOF, more is the flexibility of the arm, but as mentioned above, flexibility also depends on other factors.

Page No.
Date

Q.3) (b) DH parameter table for 1st arm is :-

Joints	θ	d	α	a
J1	0	$\approx (267/2)$	$-\pi/2$	0
J2	0	0	0	0
J3	0	0	$-\pi/2$	131
J4	0	$\approx (342.5/2)$	$\pi/2$	0
J5	0	0	$-\pi/2$	76
J6	0	97	0	0

(kept at zero, inorder to make DH-table)

Significance :-

→ Conventional kinematics study of serial manipulators involved proper selection of coordinate frames of ref. Here, the use of D-H algorithm for assigning coordinate frames helps.

Geometrically → The DH parameters (θ, d, α, a) establish standard and easy framework by defining relationship betn consecutive links/joints.
It thus specifies position and orientation of each link relative to ~~the~~ other joints.

Kinematically → These parameters helps in simplifying the forward & inverse kinematic problems by locating robots end-effector in space.

→ It gives the idea of range of end-effector position and joint's rotation.

→ Thus, these parameters are also used to model simulations of robotic arm's movements, because of the fact that these parameters reveal ~~the~~ accurately the robot's geometry and kinematics.

Q.4. A python code to obtain the (q) value is :-

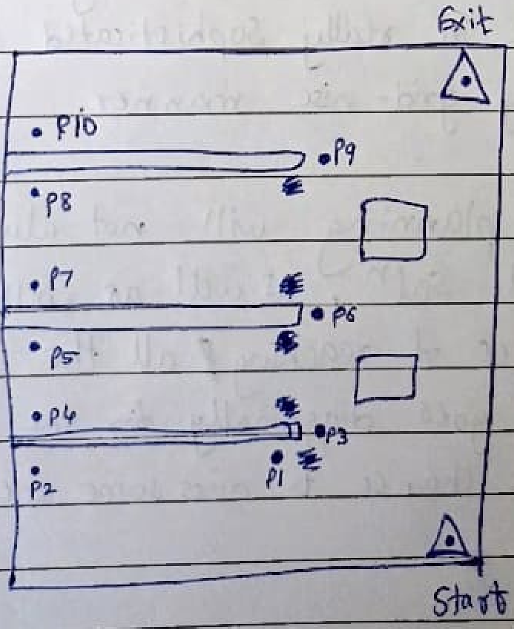
```
⇒ # Importing required libraries.
from spatialmath import SE3
from roboticstoolbox inport Puma560
# Defining end-effector coordinates.
T = np.array([[1, 0, 0, 0.5],
              [0, 1, 0, -0.1],
              [0, 0, 1, 0.5],
              [0, 0, 0, 1]])

robot = Puma560()
puma = SE3.models.Puma560()
# Calculate inverse kinematics(IK)
q = puma.ikine_LM(T)

# print the joint values.
print ("Joint values (q):")
print (q).
```

also import numpy as np

Q.5) (a) The concept of waypoints was developed in order to let the robot (algorithm in the robot) create a trajectory from the start pt. till the end pt. via surely going through some pre-defined way points which will thus be optimized according to robot speed, obstacles in the direct path from start to end, and multi-objective planning with precision & smoothness.

→ In this case, ~~the~~ obstacles are avoided (tables) and 2 goals are accomplished together
1. Scanning all books.
2. Reaching the end in short path while scanning.

→



Q.5)

(b) → In this of given library, we should use (A^* - algorithm), which will guarantee an optimal solution in this environment. Our library (given) being mostly rectangular in shape & the obstacles are also rectangular, thus this environment can be easily divided into a grid-base (2D grid), which thus also makes the book shelves in a linear row of squares inside the grid.

∵ Our library is not much complex, our computation will also be light, as well as grid-formation & finally optimal path to reach end as well as scanning can be done in really sophisticated and easy grid-wise manner.

Waypoint planning will not always produce optimal sol^n, as well as will not have assurance of reaching all the shelves [ex, if robot goes diagonally for the next waypoint, it has chance to miss some shelves.]

https://youtube.com/playlist?list=PLZaGkBteQK3HQFSWDM7-yRQWTd86DeDIY&si=WsbTAVgIMAXb4gUS

https://locusrobotics.com/what-are-autonomous-robots/

https://github.com/petercorke/robotics-toolbox-python#code-examples

https://www.youtube.com/watch?v=QR3U1dgc5RE&list=PLn8PRpmsu08rLRGrnF-S6TyGrmcA2X7kg&index=4

https://chat.openai.com/

https://www.youtube.com/watch?v=zI64DyaRUvQ&list=PLggLP4f-rq01z8VLqhDC94W2nWpWpZoMj&index=3

https://www.youtube.com/watch?v=798mAiGT-i8

https://www.roborealm.com/help/Path_Planning.php#:~:text=Waypoints%20%2D%20Add%20in%20waypoint%20coordinates,points%20from%20a%20specified%20variable.

https://www.trossenrobotics.com/xarm-6.aspx#:~:text=The%20xArm%206%20is%20a,accuracy%20of%20%2B%2F%2D%200.1mm.

https://www.youtube.com/watch?v=DPO9Se6ZqN0

https://gymkhana.iitb.ac.in/instiapp//org/the-humanoid-project