Лабораторна робота №2

ОСНОВИ ПРОГРАМУВАННЯ РОБОТА *Braccio* В СЕРЕДОВИЩІ *Arduino IDE*

Мета роботи – отримати практичні навички роботи у програмному середовищі *Arduino IDE* щодо відпрацювання основних рухів ланками маніпуляційної системи робота *Braccio.*

2.1. Теоретичні відомості

2.1.1. Основі відомості про робот моделі *TinkerKit Braccio*

TinkerKit Braccio – це робот ангулярної системи координат, маніпуляційна система (МС) якого має 5 ступенів рухомості, що реалізуються 6 рухомими ланками від L2 до L6 та затискний пристрій L8. Вказані ланки приводяться в рух відповідними двигунами від M1 до M6 (див. загальний вигляд робота на рис. 2.1):

- L1 – Deatl (основа);
- L2 – Base (основа, приводиться в рух двигуном M1 відносно ланки L1);
- L3 – Shoulder (плече, приводиться в обертовий рух двигуном M2);
- L4 – Elbow (лікоть, приводиться в обертовий рух двигуном M3);
- L5 – Vertical wrist (вертикальний зап’ясток, приводиться в обертовий рух двигуном M4);
- L6 – Rotary wrist (обертальний зап’ясток, приводиться в обертовий рух двигуном M5);

- L7 – конструктивна константа;
- L8 – Gripper (схват, затискний пристрій, приводиться в рух двигуном M6).

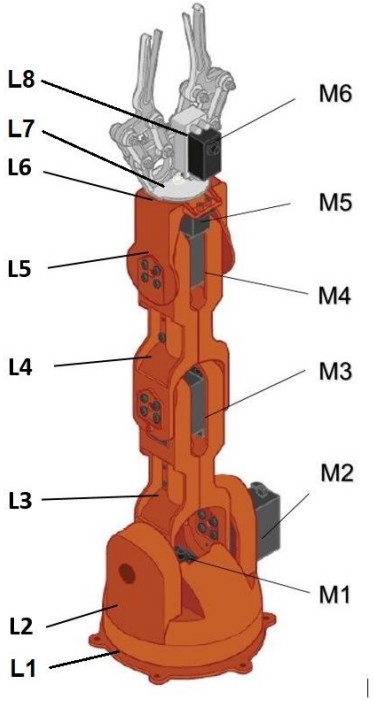


Рис. 2.1. – Загальний вигляд робота TinkerKit Braccio

Конфігурацію робота можна змінювати, зменшуючи кількість рухомих ланок його МС. На робот встановлений класичний двопальцевий схват (gripper), який при бажанні можна замінити на легкий інструмент або інший прилад, наприклад, камеру, ліхтар тощо.

Серводвигуни (рис. 2.2) забезпечують рух всіх ланок МС та роботу схвата. Потужність двигунів різна і залежить від ланки, яку двигун приводить в рух. Наприклад, на ланці горизонтального кутового переміщення станини встановлений двигун M1 потужніший, ніж на схваті (двигун M6).

МС даної моделі робота працює в ангулярній системі координат, так як ланка L2 забезпечує обертання інших ланок МС навколо вертикальної осі, а інші навколо горизонтальних осей щодо положення, зображеного на рис. 2.2. Ланки L1, L3, L4 і L5 мають кут повороту ±90°, а ланка L2 ±75°. Серводвигун схвата відпрацьовує кут в інтервалі від 0° до 70°.

Кінематична структура робота *TinkerKit Braccio* та його геометричні та кінематичні параметри вказані на рис. 2.2 та 2.3.

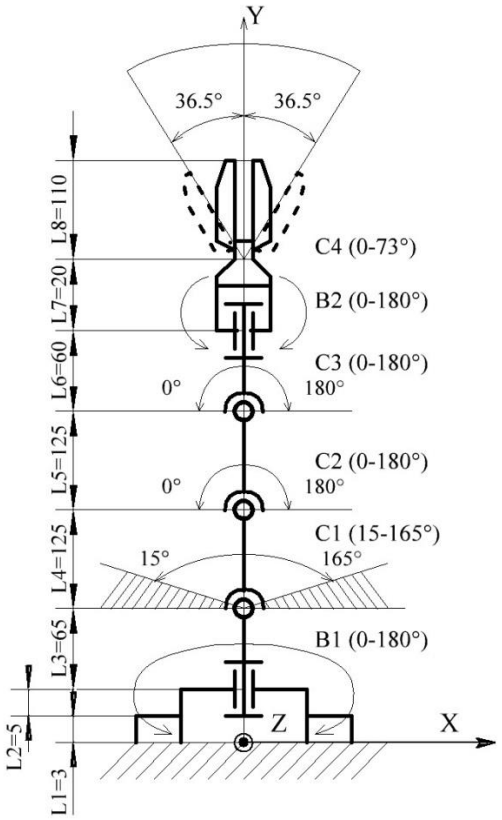


Рис. 2.2. – Кінематична структура робота TinkerKit Braccio

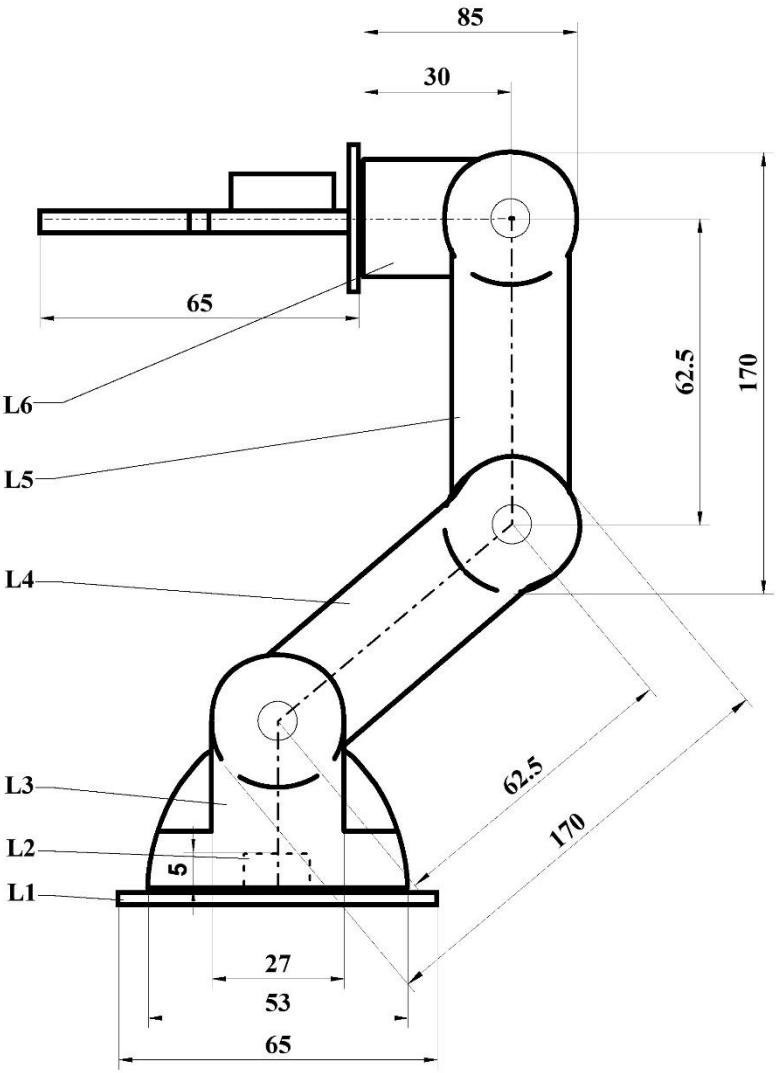


Рис. 2.3. – Основні розміри робота TinkerKit Braccio

Серводвигуни М1 – М6 розміщені на корпусах відповідних ланок та закріплені за допомогою гвинтів у спеціальних пазах.

На рис. 2.4 як тприклад показано розміщення серводвигунів М1 та М2.

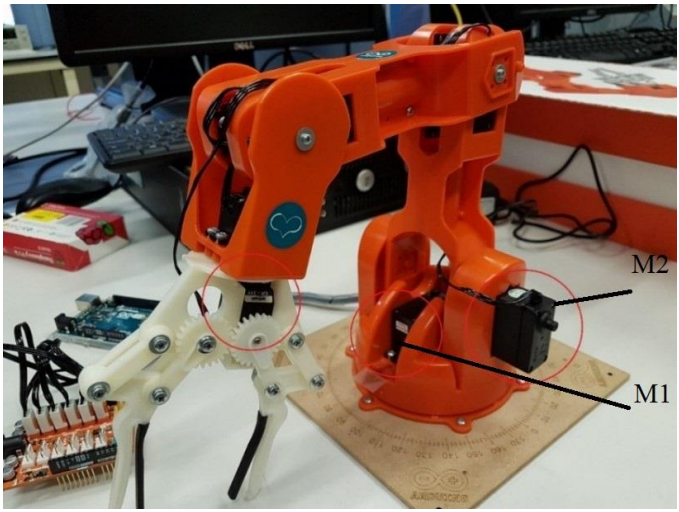


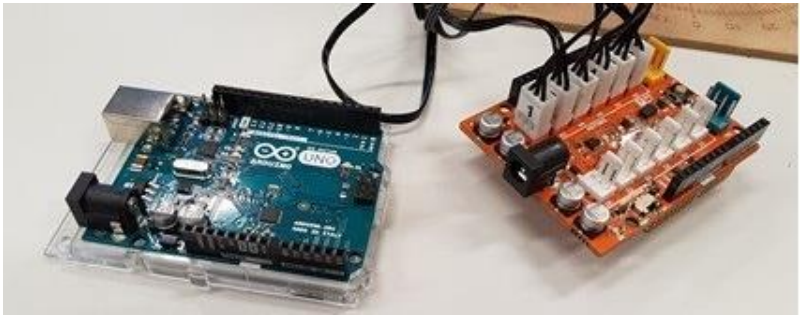
Рис. 2.4. – Розташування серводвигунів М1 та М2

Серводвигун (від лат. *Servus* – слуга, помічник, раб) – механічний привід з автоматичною корекцією стану кутового положення ротора (вала) через внутрішній негативний зворотний зв'язок відповідно до параметрів, заданих із зовні) складається із редукторної системи шестерень зі стопорами ходу (для відпрацювання кута від 0° до 180°), внутрішньої плати керування, двигуна та корпусу.

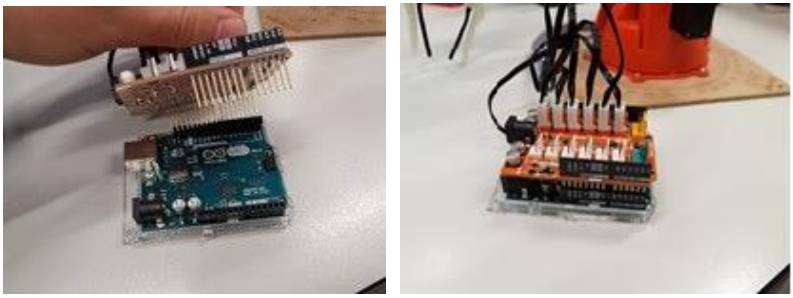
На рис. 2.5 показано внутрішню будову серводвигунів, які використовуються для переміщення ланок робота *TinkerKit Braccio*.

Робот функціонує на базі мікроконтролера *Arduino Uno* та модуля *Braccio Shield* (рис. 2.6). Для функціонування робота необхідно встановити спеціальні бібліотеки на ПЗ *Arduino IDE*. Програма стандартно компілюється на комп’ютері, а потім записується у мікроконтролер *Arduino Uno*. Модуль *Braccio Shield* може керувати 6 серводвигунами (М1 – М6) одночасно.

Рис. 2.5. – Внутрішня будова серводвигуна



а)



б) *в)*

Рис. 2.6. – Загальний вигляд мікроконтролера Arduino Uno та модуля Braccio Shield (а), під'єднання модуля Braccio Shield до Arduino Uno (б), з'єднані модуль Braccio Shield та Arduino Uno (в)

Також є можливість підключити до модуля 6 датчиків (в роз'єми *INPUTS* за рис. 2.7). Дана функція розширює можливості застосування робота.

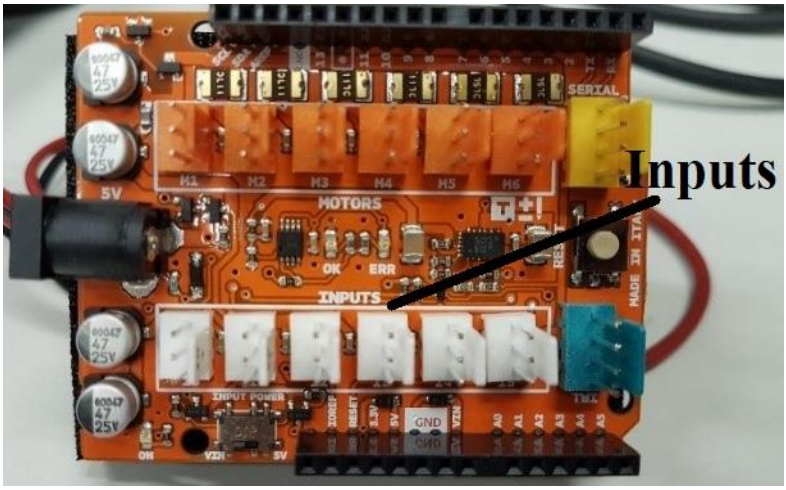


Рис. 2.7. – Загальний вигляд плати розширення Braccio Shield

Напруга на модуль керування двигунами подається з окремого блоку живлення, який забезпечує 5 В постійного струму силою до 5000 mA. Теоретично система може працювати від потужного акумулятора.

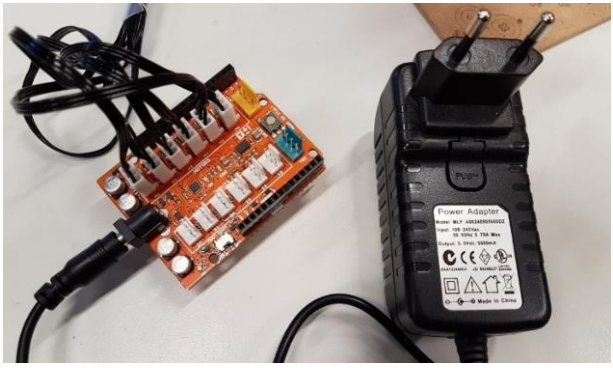


Рис. 2.8. – Модуль Braccio Shield та блок живлення модуля

2.2. Технічні характеристики робота *TinkerKit Braccio*

2.2.1. Деталі та склад

Ланки робота можна зібрати в різні конфігурації (рис. 2.9), що реалізується завдяки різним деталям та взаємозамінним модулям. В комплект поставки включені нижче приведені деталі, плати та двигуни, а саме:

• 21 пластикових деталей;
• 2 серводвигуни *SR 311*;
• 4 серводвигуни *SR 431*;
• 1 плата *Braccio shield* для *Arduino Uno*;
• 1 блок живлення 5В, 5А;
• 1 протектор кабелів.

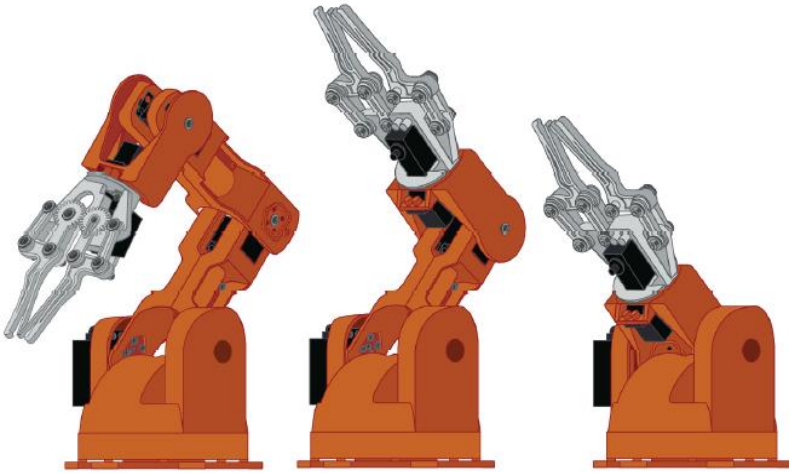


Рис. 2.9. – Можливі конфігурації побудови робота TinkerKit Braccio

Характеристики повністю зібраного робота *TinkerKit Braccio* приведені в табл. 2.1.

Таблиця 2.1. – Характеристики робота TinkerKit Braccio (у конфігурації з 6 ланками)

№	Найменування	Параметр	Величина
1	Вага	грам	792
2	Робоча відстань	см	80
3	Висота	см	52
4	Ширина основи	см	14
5	Ширина захвату	мм	90
6	Довжина кабелю	см	40
7	Вантажопідйомність при робочій відстані 32 см	грам	150
8	Максимальна маса	грам	400

2.3. Контролер *Arduino Uno*

2.3.1. Загальні відомості про контролер *Arduino Uno*

Arduino Uno – це пристрій на основі мікроконтролера *ATmega328*. У його склад входить:

- 14 цифрових входів / виходів (з них 6 можуть використовуватися в якості ШІМ-виходів; ШІМ – широтно-імпульсна модуляція), 6 аналогових входів;
- кварцевий резонатор на 16 МГц;
- роз'єм *USB*;
- роз'єм живлення;
- роз'єм *ICSP (*від англ. *In Circuit Serial Programming* - внутрішньосхемне програмування);
- кнопка скидання.

Для початку роботи з пристроєм досить просто подати живлення від *AC/DC*-адаптера або батарейки, або підключити його до комп'ютера за допомогою *USB*-кабелю.

Загальний вигляд контролера *Arduino Uno* представлено на рис. 2.10.

Рис. 2.10. – Контролер Arduino Uno

Характеристики контролера *Arduino Uno* приведені у табл. 2.2.

Таблиця 2.2. – Характеристики контролера Arduino Uno

№	Найменування	Параметр	Величина
1	Мікроконтролер	–	ATmega 328
2	Робоча напруга	В	5
3	Напруга живлення	В	7–12
4	Максимальний струм одного вивода	мА	40
5	Flash-пам’ять	КБ	32
6	SRAM	КБ	2

2.3.2. Живлення контролера *Arduino Uno*

Напруга живлення для *Arduino Uno* подається від *USB* або від зовнішнього джерела живлення.

В якості зовнішнього джерела живлення (не *USB*) може використовуватися мережевий *AC / DC* - адаптер або акумулятор /

батарея. Штекер адаптера необхідно вставити у відповідний роз'єм живлення на платі. У разі живлення від акумулятора / батареї їхні контакти необхідно під'єднати до виводів *Gnd* та *Vin* роз'єму *POWER*.

Нижче перераховані виводи живлення, розташовані на платі:

VIN **–** напруга, що надходить в *Arduino Uno* безпосередньо від зовнішнього джерела живлення (не пов'язане з 5 В від *USB* або іншим стабілізованою напругою). Через цей вивід можна як подавати зовнішнє живлення, так і споживати струм, коли пристрій живиться від зовнішнього адаптера;

5V **–** на вивід надходить напруга 5 В від стабілізатора напруги на платі незалежно від того, як живиться пристрій: від адаптера (7─12 В), від *USB* (5 В) або через вивід *VIN* (7–12 В). Живити пристрій через виводи *5V* або *3V3* не рекомендується, оскільки в цьому випадку не використовується стабілізатор напруги, що може призвести до виходу плати з ладу;

3.3V **–** 3.3 В, напруга, що надходять від стабілізатора напруги на платі. Максимальний струм, споживаний від цього виводу, становить 50 мА;

GND – вивід землі або мінус у замкнутому колі;

IOREF – цей вивід надає платам розширення інформацію про робочу напругу мікроконтролера *Arduino Uno*. Залежно від напруги, зчитаного з виводу *IOREF*, плата розширення може переключитися на відповідне джерело живлення або задіяти перетворювачі рівнів, що дозволить їй працювати як з 5 В, так і з 3.3 В пристроями.

2.3.3. Входи і виходи контролера *Arduino Uno*

Нижче представлені позначення виходів мікроконтролера *Arduino Uno*, що відповідають рис. 2.10.

З використанням функцій *pinMode (), digitalWrite ()* і *digitalRead ()* кожен з 14 цифрових виводів може працювати в

якості входу або виходу. Рівень напруги на виводах обмежений 5В. Максимальний струм, який може віддавати або споживати один вивід, становить 40 мА.

Нище представлений опис роз'ємів.

Послідовний інтерфейс: роз'єми *0* (*RX*) і ***1* (*TX*).** Використовуються для отримання (*RX*) і передачі (*TX*) даних по послідовному інтерфейсу. Ці виводи з'єднані з відповідними виводами мікросхеми *ATmega8U2*, яка виконує роль перетворювача *USB-UART* (*UART* – від англ. *Universal Asynchronous Receiver/Transmitter* – універсальний асинхронний приймач/передавач).

Зовнішні переривання: роз'єми *2* та *3*. Можуть слугувати джерелами переривань, що виникають при фронті, спаді або при низькому рівні сигналу на цих виводах. Для отримання додаткової інформації див. функцію *attachInterrupt ()*.

ШІМ: роз'єми *3, 5, 6, 9, 10* та *11*. За допомогою функції *analogWrite ()* можуть виводити 8-бітові аналогові значення в вигляді ШІМ-сигналу.

Інтерфейс *SPI*: роз'єми *10* (*SS*), *11* (*MOSI*), *12* (*MISO*), *13* (*SCK*). Із застосуванням бібліотеки *SPI* дані виводи можуть здійснювати зв'язок по інтерфейсу *SPI*.

Світлодіод: роз'єм *13*. Вбудований світлодіод, приєднаний до виводу *13*. При відправці значення *HIGH* світлодіод вмикається, при відправці *LOW* – вимикається.

В *Arduino Uno* є 6 аналогових входів (*A0* – *A5*), кожен з яких може представити аналогову напругу у вигляді 10-бітного числа (1024 різних значень). За замовчуванням вимірювання напруги виконується в інтервалі від 0 до 5 В.

Крім перерахованих на платі існує ще кілька виводів, а саме:

***AREF* –** опорна напруга для аналогових входів. Може бути задіяний функцією *analogReference ();*

***Reset* –** формування низького рівня (*LOW*) на цьому виводі призведе до перезавантаження мікроконтролера. Зазвичай цей

вивід слугує для функціонування кнопки скидання на платах розширення.

2.3.4. Програмне забезпечення *Arduino IDE*

IDE (від англ. *Integrated Development Environment* – інтегроване середовище розробки) – це додаток або група додатків (середовище), призначених для створення, налаштування, тестування та обслуговування програмного забезпечення.

Інтегроване середовище розробки характеризується наявністю складної функціональності, включаючи редагування і компіляцію вихідного коду, створення програмних ресурсів, створення баз даних і т.д.

Для програмування контролерів серії *Arduino* фірмою-розробником створено програмне забезпечення, що відповідає основним вимогам типового середовища *IDE*. Це не потужне програмне забезпечення, а проста функціональна програма, яка дозволяє писати, компілювати і завантажувати програму в мікроконтролер.

Середовище розробки *Arduino IDE* являє собою:

- текстовий редактор програмного коду;
- область повідомлень;
- вікно виведення тексту (консоль);
- панель інструментів;

\- кілька меню.

Для завантаження програм і зв'язку середовище розробки підключається до апаратної частини контролерів *Arduino*.

Програми для контролерів *Arduino* пишуться на звичайній мові програмування *C++*, доповненою простими і зрозумілими функціями для керування введенням / виведенням на контактах.

Arduino IDE містить множину попередньо створених бібліотек. Бібліотеки додають додаткову функціональність скетчам (від англ. *sketch* – ескіз, скетч, чорновик), наприклад, при роботі з апаратною частиною або при обробці даних. Одна або

кілька директив “*#include”* будуть розміщені на початку коду скетчу з подальшою компіляцією бібліотек разом зі скетчем. Завантаження бібліотек вимагає додаткового місця в пам'яті контролерів *Arduino*.

Для роботи з платами *Arduino* та модуля *Braccio Shield* необхідна спеціальна програма *Arduino IDE* (рис. 2.11). *Arduino IDE* – це середовище розробки програм для платформ на базі мікропроцесорів *Arduino* (інше визначення див. вище). За допомогою цього компілятора розроблені програми записуються у пам'ять мікропроцесора.

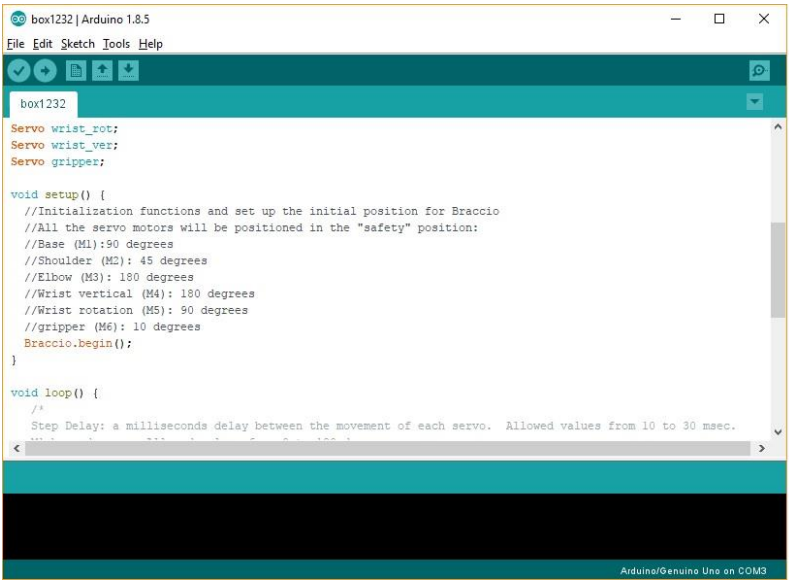


Рис. 2.11. – Вікно середовища розробки програм Arduino IDE

2.4. Попередня підготовка обладнання для роботи з середовищем програмування *Arduino IDE* та роботом *TinkerKit Braccio*

2.4.1. Встановлення *Arduino IDE*

Для початку роботи із лабораторним стендом, який описано далі в п. 2.6, необхідно встановити на комп'ютер інтегроване середовище програмування контролерів *Arduino – Arduino IDE*. Програма доступна для наступних операційних систем (ОС):

- *Windows* (починаючи з версії *XP* до *10*);
- *Linux* (*x32*, *x64* та *ARM*);
- *Mac OS X*.

Для цього необхідно перейти за посиланням: https://www.arduino.cc/en/Main/Software та завантажити інсталятор останньої версії програми.

Скріншот вікна браузера за вказаним посиланням представлений на рис. 2.12.

Рис. 2.12. – Скріншот вікна браузера за посиланням для завантаження інтегрованого середовища програмування Arduino IDE

Після вибору інсталятора для відповідної операційної системи у браузері відобразиться вікно, у якому пропонується завантаження безкоштовний інсталятор або завантаження із благодійним внеском (в буквальному розумінні).

Обирається пункт “*just download*”, що показано на рис. 2.13 і виділено красною рамкою у нижньому правому куті, для отримання інсталятора без благодійного внеску.

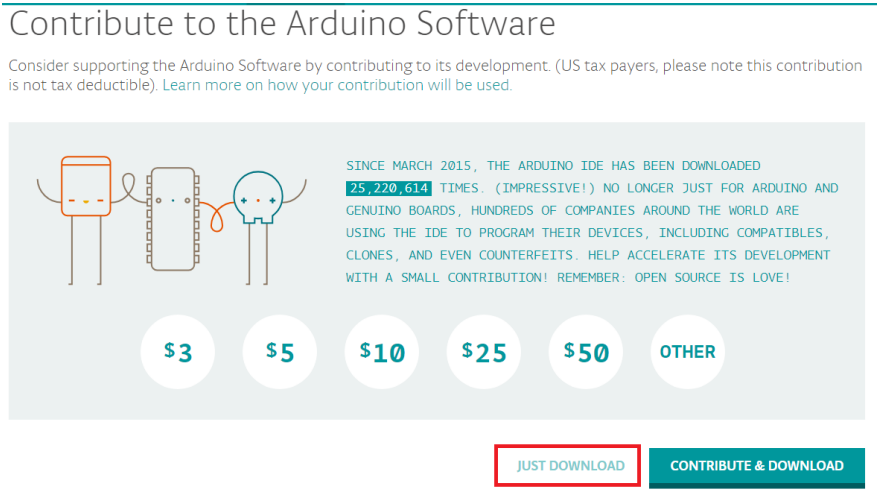


Рис. 2.13. – Скріншот вікна завантаження інсталятора

Після натискання кнопки “*just download*” обирається папка розміщення інсталятора, щоб при завершенні завантаження знайти файл та запустити його для встановлення *Arduino IDE*.

На прикладі рис. 2.14 обрано папку “*Робочий стіл*”.

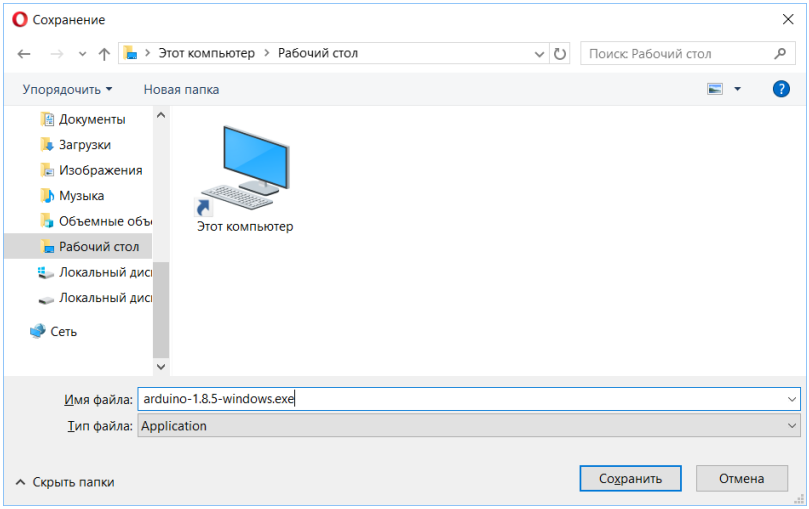


Рис. 2.14. – Скріншот вікна завантаження інсталятора

При запуску інсталятора відобразиться вікно (рис. 2.15), на якому представлені ліцензійні права розробника додатка *Arduino IDE*.

Встановлення програмного середовища виконується натисканням кнопки “*I Agree*”, що виділено червоною рамкою у правому нижньому куті на рис. 2.15.

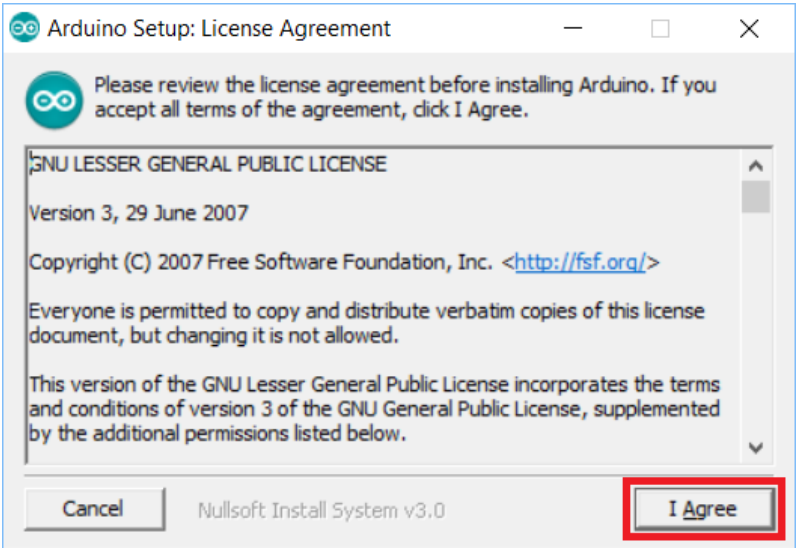


Рис. 2.15. – Скріншот вікна погодження з ліцензійними правами Arduino IDE

Наступний крок – вибір потрібних компонентів додатку *Arduino IDE* для встановлення їх на комп'ютер. Для цього виділяються позначками всі компоненти, як показано на рис. 2.16, і натискається кнопка "*Next*", що виділено на рис. 2.16 красною рамкою у правому нижньому куті.

Далі обирається папка, в яку буде встановлено програмне забезпечення. Для прикладу на рис. 2.17 показаний стандартний шлях встановлення програмного забезпечення на локальному диску *C:*. Він виконується натисканням кнопки "*install*", що виділено на рис. 2.17 красною рамкою у правому нижньому куті.

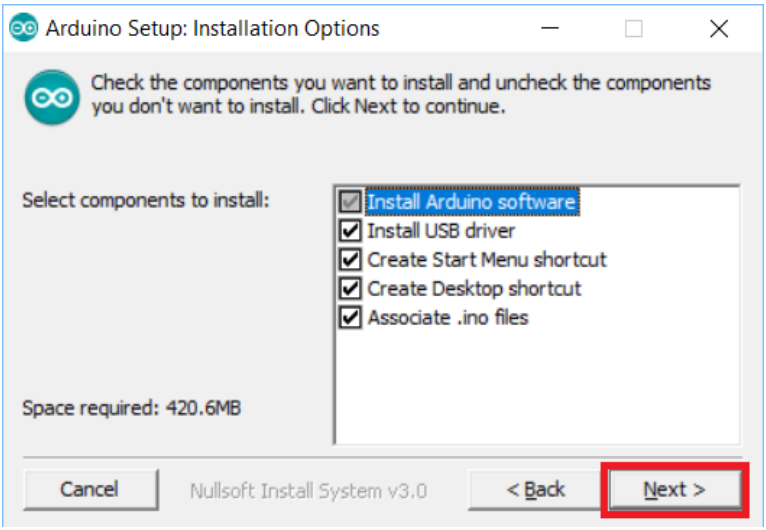


Рис. 2.16. – Скріншот вікна вибору компонентів Arduino IDE для їх встановлення

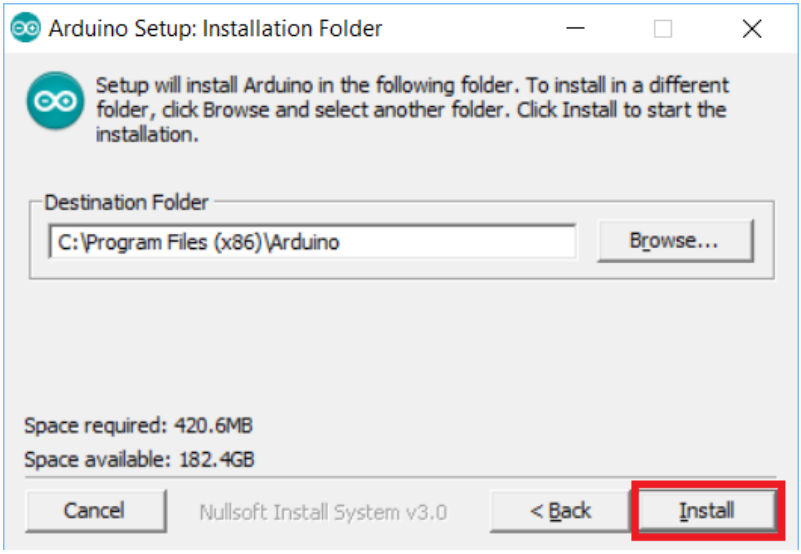


Рис. 2.17. – Скріншот вікна вибору папки для встановлення програми Arduino IDE

Під час встановлення Arduino IDE комп’ютер може запросити доступ на встановлення драйверів для портів. За такої ситуації необхідно надати дозвіл програмі на їх встановлення.

На рис. 2.18 показано завершення встановлення додатку, і завершення роботи з інсталятором натисканням кнопки “Close”.

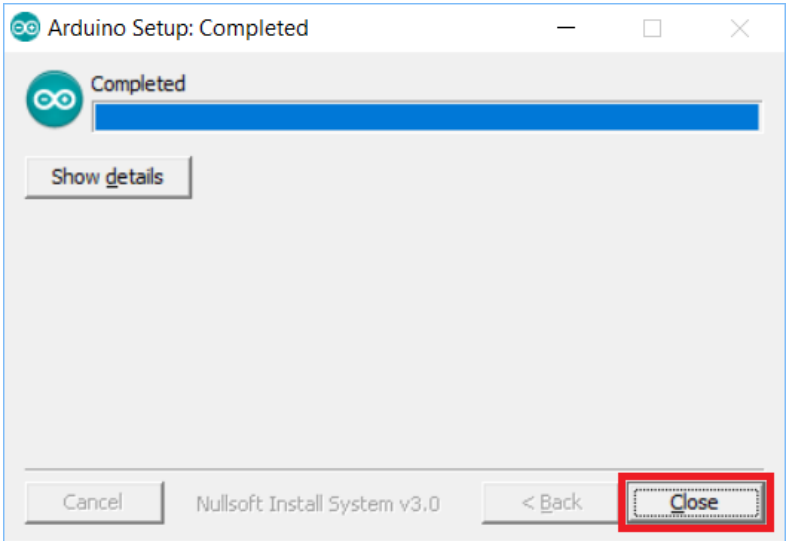


Рис. 2.18. – Скріншот завершення встановлення Arduino IDE

Після виконання попередніх кроків на робочому столі має з’явитись ярлик “*Arduino*”, який показано на рис. 2.19.

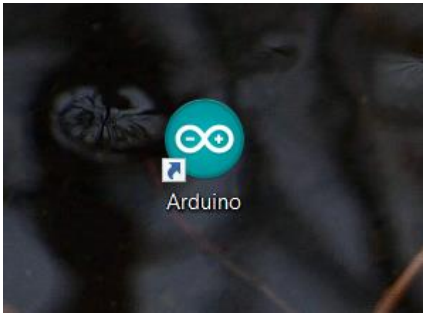


Рис. 2.19. – Ярлик додатку Arduino IDE

2.4.2. Запуск *Arduino IDE*

Після завантаження та встановлення додатку *Arduino IDE* його запуск виконується подвійним кліком миші на ярлику ”*Arduino*”, після чого з’явиться вікно *Arduino IDE* (рис. 2.19).

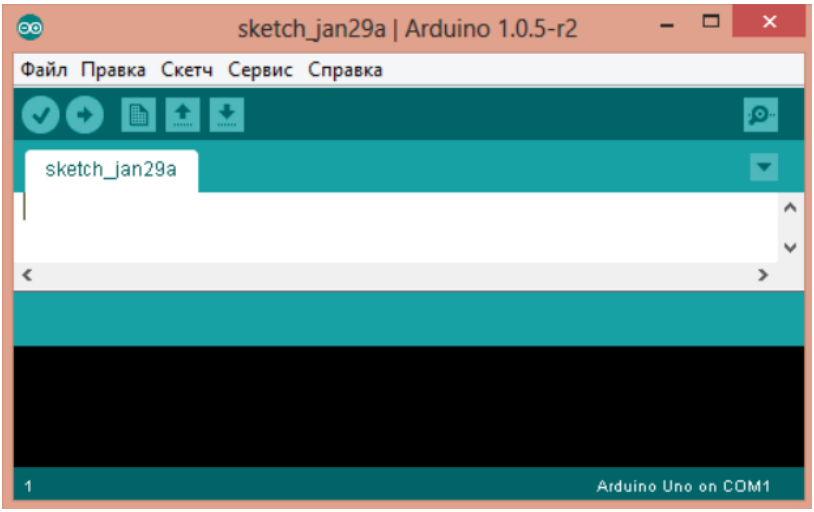


Рис. 2.19. – Вікно додатку Arduino IDE

2.4.3. Підключення *Arduino Uno* до комп’ютера

Після встановлення додатку *Arduino IDE* потрібно підключити *Arduino Uno* до комп'ютера.

Для цього необхідно з’єднати *Arduino Uno* з комп'ютером через *USB*-кабель. На платі *Arduino Uno* загориться світлодіод “*ON*” і почне блимати світлодіод “*L*” (рис. 2.20). Це означає, що на плату подано живлення і мікроконтролер *Arduino Uno* почав виконувати “*прошиту*” за замовчуванням програму “*Blink*” (миготіння світлодіодом).

Рис. 2.20. – Сигнали про живлення Arduino UNO

Для налаштування *Arduino IDE* на роботу з *Arduino Uno* необхідно знати, який номер COM-порту присвоїв комп'ютер *Arduino Uno*. Для цього потрібно зайти в "*Диспетчер пристроїв Windows*" (рис. 2.21) і розкрити вкладку "*Порти (COM* та *LPT)*".

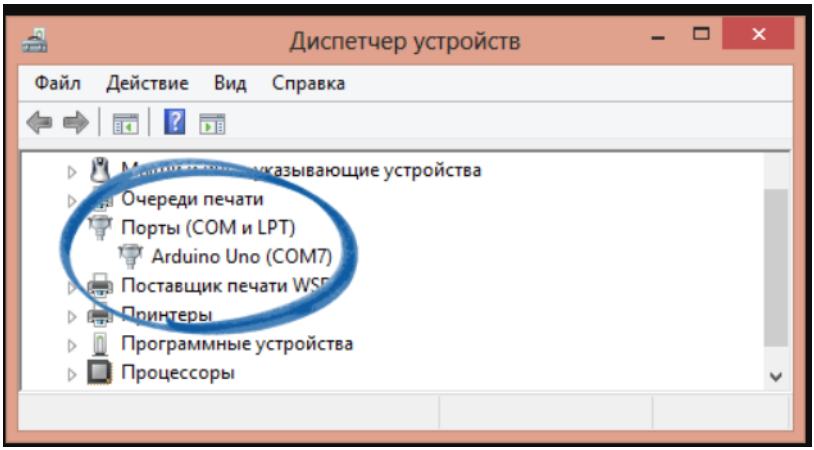


Рис. 2.21. – Диспетчер пристроїв Windows

При цьому операційна система розпізнала плату *Arduino Uno* як *COM*-порт, підібрала для неї правильний драйвер і призначила цього *COM*-порту номер *7*. Якщо підключити до комп'ютера іншу плату *Arduino*, то операційна система призначить їй інший номер. Тому при підключені декількох плат *Arduino* дуже важливо не заплутатися в номерах *COM*-портів.

2.4.4. Налаштування *Arduino IDE* на роботу з *Arduino Uno*

Для вказаного в заголовку необхідно забезпечити знаходження плати *Arduino Uno* в середовищі *Arduino IDE* в його *COM*-порту "*COM7*".

Для цього необхідно перейти в меню "*Сервіс*" → *"Послідовний порт"* і обрати порт "*COM7*" (рис. 2.22). Тепер *Arduino IDE* "знає", що плата знаходиться на порту "*COM7*".

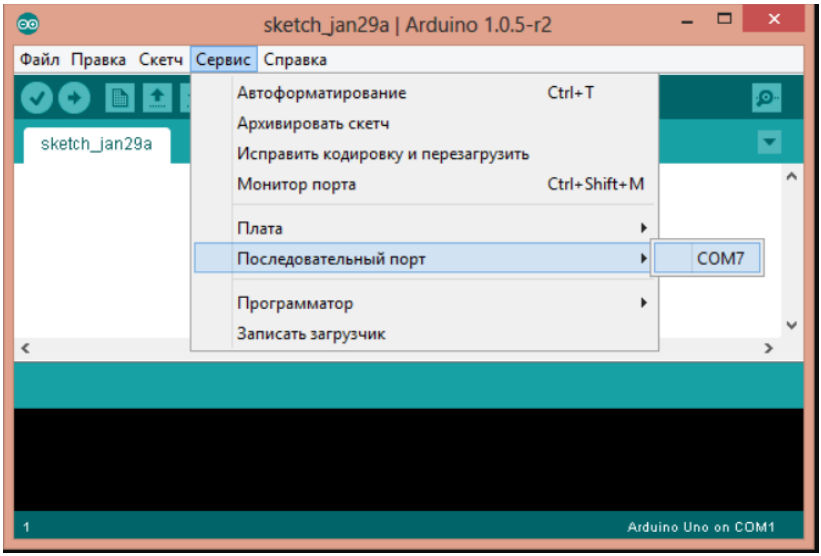


Рис. 2.22. – Вибір COM-порту

Для правильної компіляції коду необхідно встановити, що середовище програмування *Arduino IDE* працює з *Arduino Uno*.

Для цього перейти в меню “*Сервіс*” → “*Плата*” і вибрати плату “*Arduino Uno*” (рис. 2.23).

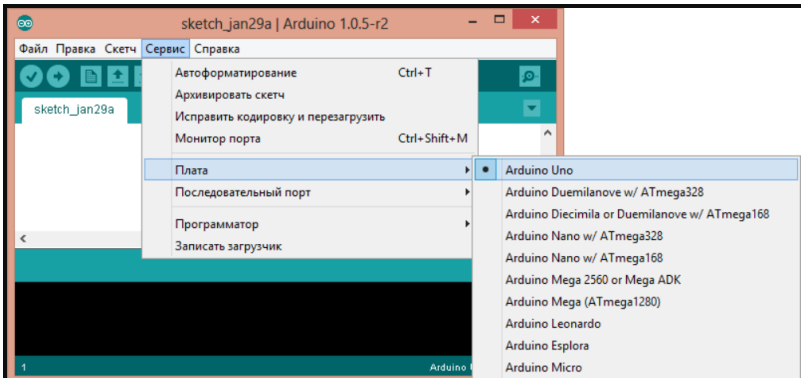


Рис. 2.23. – Вибір плати Arduino UNO

2.4.5. Завантаження першого скетча

Arduino IDE містить готові приклади попередньо розі’язаних задач для можливого швидкого інтегрування даних прикладів в основну програму.

Для завантаження першої програми необхідно вибрати скетч під назвою “*Blink*” (рис. 2.24).

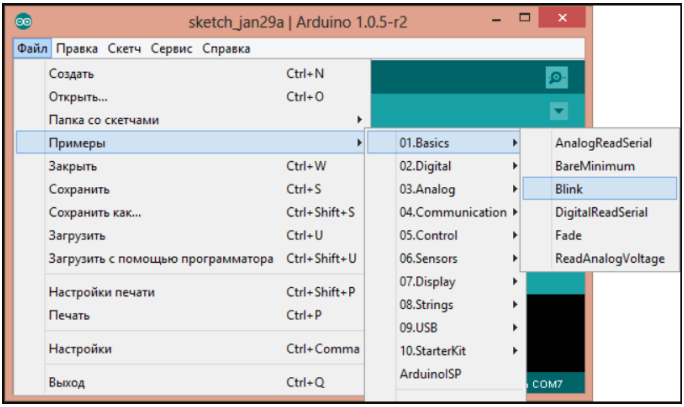


Рис. 2.24. – Вибір програми “Blink”

Для завантаження програми “*Blink*” необхідно натиснути кнопку “*Завантажити*” (рис. 2.25).

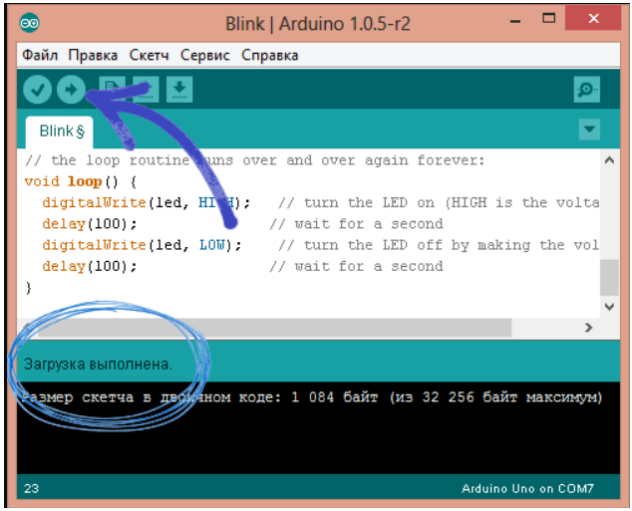


Рис. 2.25. – Завантаження програми “Blink”

2.4.6. Встановлення бібліотеки *Braccio*

Для забезпечення функціонування стенду (див. далі рис 2.31) та керування його складовими необхідно використовувати бібліотеку “*Braccio*”, для завантаження якої необхідно перейти за посиланням https://github.com/arduino-org/arduino-library-braccio (рис. 2.26).

Для підключення завантаженої бібліотеки “*Braccio*” необхідно виконати наступні дії: “*Скетч*” → “*Підключити бібліотеку*” → “*Добавити .ZIP бібліотеку…*” (рис. 2.27).

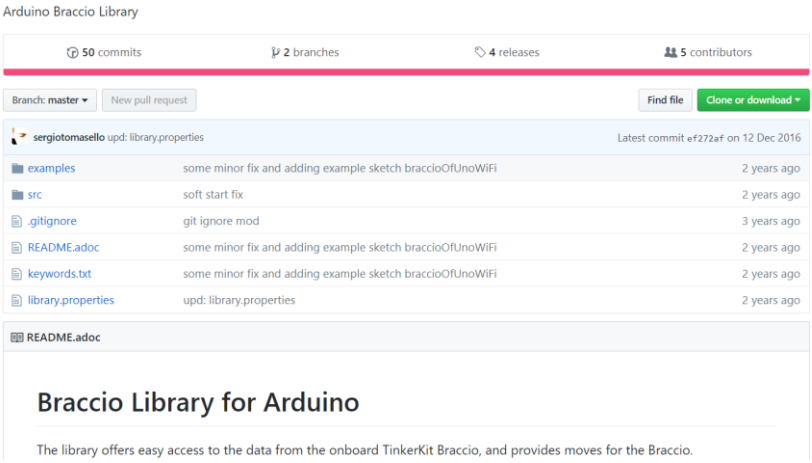


Рис. 2.26. – Ресурс з бібліотекою "Braccio" для TinkerKit Braccio

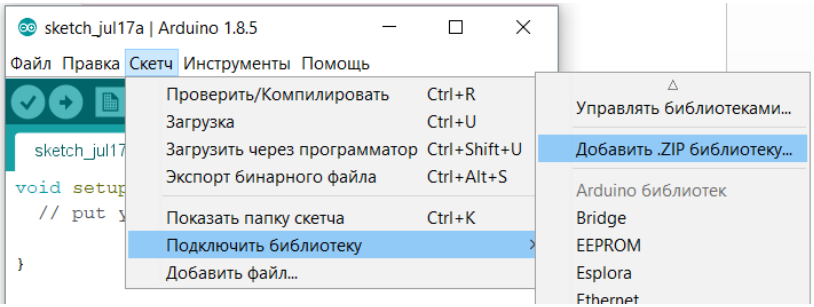


Рис. 2.27. – Підключення бібліотеки "braccio" для TinkerKit Braccio

Програмний код має стандартну структуру лише в частині, де кожному двигуну надається своє значення, а синтаксис відрізняється.

Приклад скетчу на просте переміщення в циклі з коментарями показано на рис. 2.28. В червоній рамці виділені коментарі, а в структурі *Braccio.ServoMovement* прописані значення кутів двигунів та затримки роботи між двигунами.

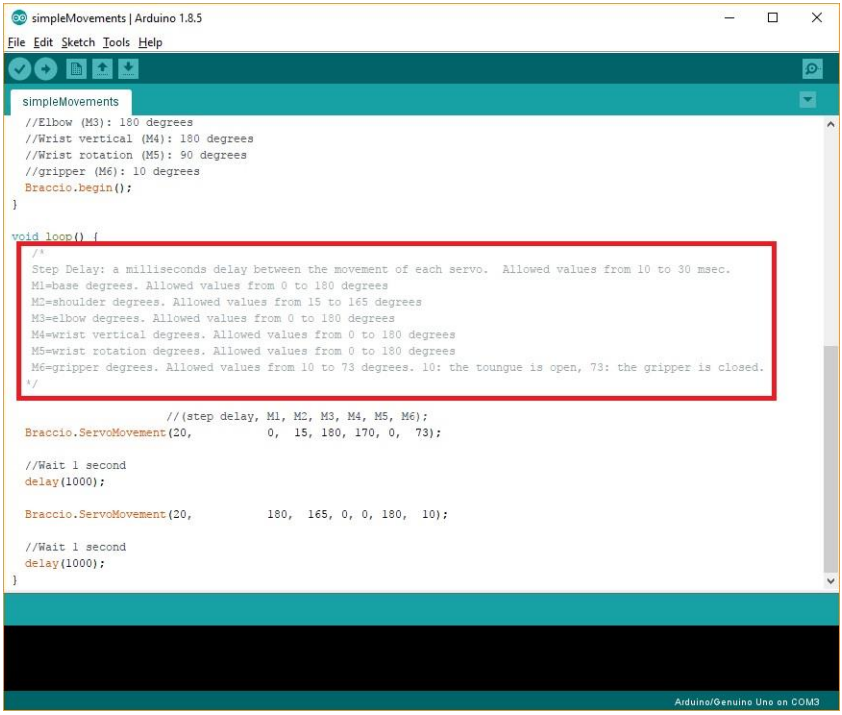


Рис. 2.28. – Приклад простого коду для робота

2.5. Структура та зміст програми

Програма відпрацювання переміщень ланок робота складається з декількох складових:

- підключення бібліотек (*#include<ім'я файлу>*);
- функція ініціалізації (*void setup ()*);
- основна функція (*void loop()*).

За допомогою команди ***#include<ім'я файлу>*** виконується підключення двох допоміжних бібліотек (<***Braccio.h***> та <***Servo.h***>). Бібліотека <***Braccio.h***> забезпечує коректну роботу плати *Arduino Uno* з платою драйвером *Braccio Shield* та передачу керуючих ШІМ-сигналів до серводвигунів. Для коректної роботи і точного відпрацювання кутів серводвигунами необхідно підключити бібліотеку <***Servo.h***>.

У функції ініціалізації ***void setup()*** ініціалізується і запускається основна функція з бібліотеки <***Braccio.h***> за допомогою команди ***Braccio.begin()***. При спрацюванні цієї функції починається подача живлення і керуючих ШІМ-сигналів до серводвигунів робота.

Основна функція ***void loop ()*** забезпечує виконання програмного коду в циклі, тобто програма буде повторно виконуватись до завершення. Для відпрацювання конкретних кутів окремих ланок робота у функції ***void loop ()*** використовується структура ***Braccio.ServoMovement(…).*** У даній структурі першою прописується тривалість затримки кроку, яка визначає кількість мілісекунд між покроковим відпрацюванням кута кожного серводвигуна. Наступні значення визначають кути повороту вала кожного серводвигуна, починаючи з М1 і завершуючи М6.

2.6. Структура та складові лабораторного стенду

Лабораторний стенд складається з таких складових груп:

- елементи управління;
- виконавчі механізми;
- допоміжні компоненти.

В групу ***елементи управління*** входять наступні компоненти:

- плати *Arduino Uno* (2 шт.);
- драйвери *Braccio Shield* (2 шт.);
- панель управління (для ручного керування 1шт.).

До групи ***виконавчі механізми*** входить два робота моделі *TinkerKit Braccio*. Опис роботів представлений в п. 2.1.1.

Група ***допоміжні компоненти*** складається з ТО (технологічних об’єктів) і станини, на якій встановлені роботи і ***елементи управління.***

На рис. 2.29 зображено лабораторний стенд з розміщенням його складових елементів.

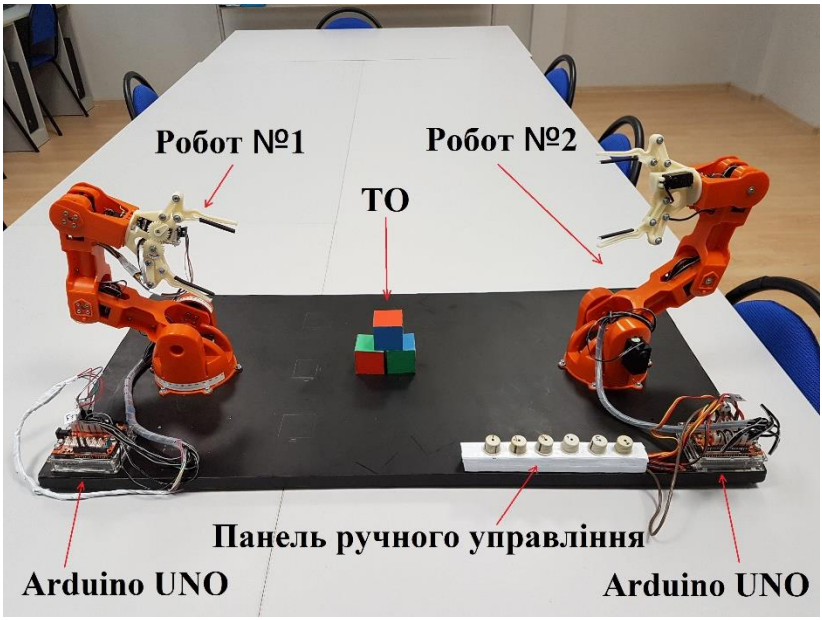


Рис. 2.29. – Загальний вигляд лабораторного стенду

Виконавчі механізми стенду працюють за керуючими сигналами, які надходять з ***елементів управління***, а саме підсилений за допомогою драйверів, керуючий сигнал від контролера *Arduino Uno* до серводвигунів.

Приведення стенду в робочий стан складається з наступних кроків:

- розробка програми;
- завантаження програми у контролер;
- підключення компонентів до джерела живлення;
- запуск і перевірка роботи програми.

Під час кроку ***розробка програми*** розробляється код траєкторії рухів ланок МС та перевіряється правильність написання програми у *програмному середовищі* ***Arduino IDE.*** Виконується компіляція коду.

Наступним кроком є ***завантаження програми у контролер***, під час якого завантажується код у контролер з ПК через порт *USB.*

Підключення компонентів до джерела живлення передбачає встановлення роботів у безпечне положення та підключення блоків живлення до мережі.

УВАГА! НЕ ДОПУСКАЄТЬСЯ ОДНОЧАСНЕ ЖИВЛЕННЯ ПЛАТИ *Arduino Uno* ЧЕРЕЗ *USB* ТА СПЕЦІАЛЬНОГО РОЗ'ЄМУ!

Останнім кроком є ***запуск і перевірка програми.*** Результатом є демонстрація правильності роботи програми за варіантом індивідуального завдання.

2.7. Приклад виконання завдання та програми простих переміщення ланок МС робота *TinkerKit Braccio*

2.7.1. Завдання

Запрограмувати переміщення ланок робота зі стартової (початкової) позиції в кінцеву для абстрактного варіанту ***:

Варіант №	Кути ланок стартової позиції							Кути ланок кінцевої позиції					
	M1	M2	M3	M4	M5	M6		M1	M2	M3	M4	M5	M6
***	0	15	180	170	0	73		180	165	0	0	180	10

2.7.2. Загальні рекомендації

Загальна схема виконання завдання включає наступні основні етапи Е:

Е1. Встановлення та запуск додатку, бібліотек та ініціалізація двигунів під відповідну бібліотеку (див. далі кроки К1 – К5 п. 2.7.3);

Е2. Запис стартового, поточного та кінцевого положень ланок МС робота (кроки К6 – К8 п. 2.7.3);

Е3. Складання коду програми за варіантом індивідуальних завдань;

Е4. Підключення плати *Arduino Uno* до ПК та завантаження програми в нього;

Е5. Перевірка працездатності складеного коду програми на лабораторному стенді (крок К11).

2.7.3. Виконання завдання

Завдання виконується наступною послідовністю кроків **К1… К11**:

К1. запустити додаток (програмне середовище *Arduino IDE)*;

К2. створити новий скетч (файл > новий, або *Ctrl + N* тобто створення нового скетча);

К3. підключити бібліотеки за допомогою команди (*#include*< ***Braccio.h*** > та *#include*< ***Servo.h*** >);

К4. ініціалізувати окремі серводвигуни під бібліотеку "*Servo.h*" за допомогою команди *Servo*:

Servo base; // серводвигун М1, ланка L2 (основа);

Servo shoulder; // серводвигун М2, ланка L3 (плече);

Servo elbow; // серводвигун М3, ланка L4 (лікоть);

Servo wrist_rot; // серводвигун М4, ланка L5 (вертикальний зап'ясток);

Servo wrist_ver; // серводвигун М5, ланка L6 (обертальний зап'ясток);

Servo gripper; // серводвигун М6, ланка L8 (схват);

К5. ініціалізувати бібліотеку "*Braccio.h*" за допомогою функції ***void setup ()*** та команди ***Braccio.begin();***

К6. записати стартове положення (стартову позицію) ланок маніпулятора робота у функції ***void loop ()*** за допомогою команди ***Braccio.ServoMovement* (20, 90, 90, 90, 90, 90, 73);**

та задати затримку в 1 секунду за допомогою команди ***delay*(1000);**

К7. записати початкове положення під №1 ланок маніпуляційної системи робота у функції ***void loop ()*** за допомогою команди:

Braccio.ServoMovement **(20, 0, 15, 180, 170, 0, 73);**

та задати затримку в 1,5 секунди за допомогою команди ***delay*****(1500);**

К8. записати кінцеве положення під №2 ланок маніпуляційної системи робота у функції ***void loop ()*** за допомогою команди:

Braccio.ServoMovement **(20, 180, 165, 0, 0, 180, 10);** та задати затримку в 0,5 секунди за допомогою команди ***delay*****(500);**

К9. підключити плату *Arduino UNO* до ПК та обрати відповідний *COM*-порт;

К10. завантажити складену за варіантом індивідуального завдання програму в *Arduino UNO*;

К11. перевірити роботу програмного коду на стенді.

Приклад частини програмного коду має вигляд:

Braccio.ServoMovement(20, 0, 15, 180, 170, 0, 73);

Це означає:

- 20 мілісекунд затримка кроку;
- M1 відпрацьовує кут 0°;
- M2 відпрацьовує кут 15°;
- M3 відпрацьовує кут 180°;
- M4 відпрацьовує кут 170°;
- M5 відпрацьовує кут 0°;
- M6 відпрацьовує кут 73°.

Нижче представлений програмний код для прикладу, у якому виконується відпрацювання роботом трьох позицій, що умовно названі “стартова”, “кобра” та “обернена кобра”. Символ “//” активує рядок коментаря, який не впливає на програмний код, а має описовий, інформативний характер.

Загальний вид процюючого програмного коду наступний:

```
#include <Braccio.h> //  команда #include <Braccio.h>,
                         підключає бібліотеку для роботи з
                         роботами
#include <Servo.h> //    команда #include <Servo.h, підключає
                         бібліотеку для роботи з серводвигунами

// ініціалізація окремих серводвигунів під бібліотеку

Servo base; // серводвигун M1 ланки L2 (база)
Servo shoulder; // серводвигун  M2 ланки L3 (плече)
Servo elbow; // серводвигун M3 ланки L4 (лікоть)
Servo wrist_rot; //  серводвигун M4 ланки L5 (вертикальний
                     зап'ясток)
Servo wrist_ver; //  серводвигун M5 ланки L6 (обертальний
                     зап'ясток)
Servo gripper; // серводвигун M6 ланки L8 (схват)

void setup() {
  // Ініціалізація функцій і встановлення ланок робота в
     початкову позицію для бібліотеки Braccio ()
  // Всі серводвигуни будуть встановлені в "безпечну"
     позицію:
  // База – ланка L2, двигун (M1): 90 градусів
  // Плече – ланка L3, двигун (M2): 45 градусів
  // Ланка – L4, двигун (M3): 180 градусів
  // Ланка – L5, двигун (M4): 180 градусів
  // Кисть – ланка L6, двигун (M5): 90 градусів
  // Схват – (gripper) ланка L8, двигун (M6): 10 градусів

  Braccio.begin(); // запуск допоміжної функції
}
// void loop() – це основна функція програми, яка буде
виконуватись в циклі
```

(повторюватись після завершення виконання)

```
void loop() {
```

/* коментар починається зі знаку “/*” та закінчується знаком “*/”; між даними знаками немає необхідності починати кожен рядок коментарів зі знаку “//”

Затримка кроку: кількість мілісекунд між покроковим відпрацюванням кутів валом кожного серводвигуна.

Допустипі значення від 10 до 30 мілісекунд із кроком 1 мілісекунда.

M1=База, ланка L2. Допустимі значення від 0 до 180 градусів

M2=Плече, ланка L3. Допустимі значення від 15 до 165 градусів

M3=Ланка L4. Допустимі значення від 0 до 180 градусів

M4=Ланка L5. Допустимі значення від 0 до 180 градусів

M5=Кисть, ланка L6. Допустимі значення від 0 до 180 градусів

M6=Схват, ланка L8. Допустимі значення від 10 до 73 градусів. (10 - схват відкритий, 73 - схват закритий).

*/

// встановлення ланок маніпуляційної системи робота у "стартове" вертикальне положення всіх ланок, схват закритий

```
//(Затримка кроку, M1, M2, M3, M4, M5, M6);
Braccio.ServoMovement(20,           90, 90, 90, 90, 90, 73);

//Затримка 1 секунда
delay(1000); // затримка в мс (1с = 1000 мс) між блоками
```

// команд до затримки та після.

// Рекомендовано запрограмувати тривалість затримки //достатньою для сприйняття кожним студентом (бригадою //студентів) відпрацювання команд, наприклад, 5, 10 секунд //тощо. Можливий інтервал програмних затримок: 0 – 32767 //мс (за технічною документацією Arduino Uno).

// встановлення ланок маніпуляційної системи робота у положення #1 "кобра" (термін умовний), схват закритий

```
//(Затримка кроку, M1, M2, M3, M4, M5, M6);
```

```
  Braccio.ServoMovement(20,          0,  15, 180, 170, 0,  73);
  //Затримка 1,5 секунди
  delay(1500);
  // Див. вище
  // встановлення ланок маніпуляційної системи робота у положення #2 "обернена кобра"(термін умовний), схват закритий
  //(Затримка кроку, M1, M2, M3, M4, M5, M6);
  Braccio.ServoMovement(20,          180,  165, 0, 0, 180,  10);

  //Затримка 0,5 секунди
  delay(500);
}
//Див. вище
```

Вказане вище має екранну форму, що подана на рис. 2.30.

```
Example_UA §
#include <Braccio.h> // <-- команда #include <Braccio.h>, підключає бібліотеку для роботи з роботами
#include <Servo.h> // <-- команда #include <Servo.h, підключає бібліотеку для роботи з серводвигунами

// ініціалізуємо окремі серводвигуни під бібліотеку //

Servo base; // серводвигун M1 ланка L1 (база)
Servo shoulder;// серводвигун  M2 ланка L2 (плече)
Servo elbow;// серводвигун M3 ланка L3
Servo wrist_rot;// серводвигун M4 ланка L4
Servo wrist_ver;// серводвигун M5 ланка L5 (кисть)
Servo gripper;// серводвигун M6 ланка L6 (схват)

void setup() {
  // Ініціалізація функцій і встановлення ланок робота в початкову позицію для бібліотеки Braccio ()
  // Всі серводвигуни будуть встановлені в "безпечну" позицію:
  // База ланка L1 двигун (M1):90 градусів
  // Плече ланка L2 двигун (M2): 45 градусів
  // Ланка L3 двигун (M3): 180 градусів
  // Ланка L4 двигун (M4): 180 градусів
  // Кисть ланка L5 двигун (M5): 90 градусів
  // Схват (gripper) ланка L6 двигун (M6): 10 градусів

  Braccio.begin(); // запуск допоміжної функції
}
// void loop() це основна функція програми яка буде виконуватись в циклі
(повторюватись після завершення виконання)
void loop() {
  /*
   Затримка кроку: кількість мілісекунд між покрокового відпрацювання кута кожного серводвигуна.
   Допустипі значення від 10 до 30 мілісекунд.
   M1=База ланка L1. Допустимі значення від 0 до 180 градусів
   M2=Плече ланка L2. Допустимі значення від 15 до 165 градусів
   M3=Ланка L3. Допустимі значення від 0 до 180 градусів
   M4=Ланка L4. Допустимі значення від 0 до 180 градусів
   M5=Кисть ланка L5. Допустимі значення від 0 до 180 градусів
   M6=Схват ланка L6. Допустимі значення від 10 до 73 градусів. (10: схват відкритий, 73: схват закритий)
  */

  // встановлення маніпуляційної системи робота у "стартове" вертикальне положення, схват закритий
  ////////////////////////(Затримка кроку, M1, M2, M3, M4, M5, M6);
  Braccio.ServoMovement(20,             90, 90, 90, 90, 90, 73);

  //Затримка 1 секунда
  delay(1000);

  // встановлення маніпуляційної системи робота у положення #1 "кобра", схват закритий
  ////////////////////////(Затримка кроку, M1, M2, M3, M4, M5, M6);
  Braccio.ServoMovement(20,           0,  15, 180, 170, 0,  73);

  //Затримка 1,5 секунди
  delay(1500);

  // встановлення маніпуляційної системи робота у положення #2 "обернена кобра", схват закритий
  ////////////////////////(Затримка кроку, M1, M2, M3, M4, M5, M6);
  Braccio.ServoMovement(20,           180,  165, 0, 0, 180,  10);

  //Затримка 0,5 секунди
  delay(500);
```

Рис. 2 30. – Екранна форма програмного модуля представленого вище прикладу

2.8. Порядок виконання роботи

1. Вивчити теоретичні відомості.

2. Виконати попередню підготовку обладнання як вказано в п. 2.4.

3. Запустити додаток *Arduino IDE* та підключити плату *Arduino Uno* до комп'ютера за допомогою *USB* кабелю.

Схема з'єднання комп'ютера із контролером *Arduino Uno* представлена на рис. 2.31.

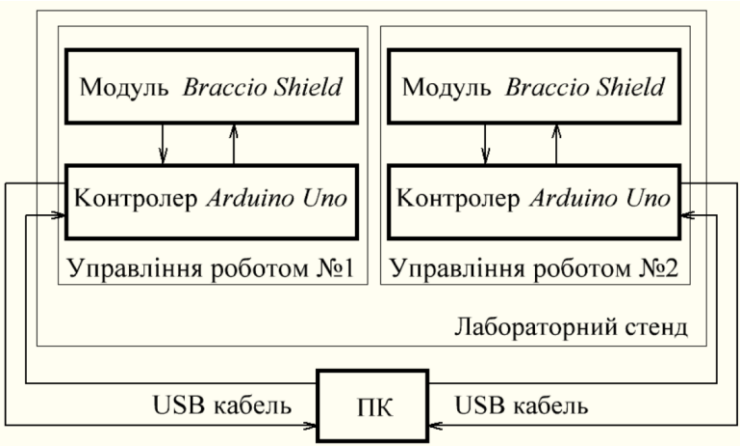


Рис. 2.31. – Схема з'єднання комп'ютера із контролером Arduino Uno

4. Написати програму переміщення ланок МС у циклі згідно із варіантом індивідуальних завдань, вказаних в п. 2.11.

5. Завантажити програму на плату *Arduino Uno* та від'єднати її від ПК.

6. Поставити ланки робота у безпечне положення та подати живлення на *Braccio Shield* за допомогою блока живлення через відповідний роз'єм.

7. Продемонструвати відпрацювання переміщення ланок МС робота, заданих варіантом.

8. Зняти відео та завантажити його в хмарне сховище (наприклад, в Google drive).

9. Створити QR-посилання на відео.

10. Оформити зміст звіту.

2.9. Зміст звіту

1. Назва та мета роботи.

2. Короткі теоретичні відомості щодо робота моделі *TinkerKit Braccio*.

3. Основні відомості щодо контролера *Arduino Uno.*

4. Схема, склад лабораторного стенду та опис його роботи.

5. Основні відомості про структуру та зміст програми.

6. Індивідуальне завдання за варіантом лабораторної роботи.

7. Програма за варіантом лабораторної роботи.

8. Представити QR–посилання відеодемострації працездатності коду за варіантом індивідуальних завдань.

9. Висновки по роботі.

2.10. Контрольні питання

1. Надати короткий опис робота *Braccio TinkerKit*.

2. Накреслити структурну схему робота *Braccio TinkerKit*

3. Склад серводвивигуна. Принцип керування серводвигуном.

4. Поняття *Arduino IDE*.

5. Короткий опис контролера *Arduino Uno*.

6. Опис входів і виходів контролера *Arduino Uno*.

7. Опис складових програми для робота за вказаним вище завданням.

8. Основні складові лабораторного стенду.

9. Стислий опис функціонування лабораторного стенду.

2.11. Варіанти індивідуальних завдань

Варіант №	Кути ланок стартової позиції						Кути ланок кінцевої позиції					
	M1	M2	M3	M4	M5	M6	M1	M2	M3	M4	M5	M6
1	10	90	30	10	90	10	160	90	110	50	180	60
2	95	90	85	60	90	10	0	90	130	90	180	60
3	20	90	50	90	90	10	180	90	120	0	180	60
4	35	90	10	85	90	10	145	90	90	0	180	60
5	70	90	80	45	90	10	180	90	130	125	180	60
6	120	90	120	60	90	10	10	90	0	120	180	60
7	30	90	20	75	90	10	160	90	100	160	180	60
8	160	90	60	30	90	10	15	90	95	0	180	60
9	45	90	110	70	90	10	90	90	60	15	180	60
10	60	90	180	160	90	10	180	90	90	40	180	60
11	60	90	130	110	90	10	0	90	50	80	180	60
12	55	90	15	95	90	10	155	90	110	0	180	60
13	110	90	65	45	90	10	20	90	150	135	180	60
14	170	90	95	120	90	10	10	90	0	40	180	60
15	0	90	100	80	90	10	180	90	180	120	180	60
16	150	90	75	150	90	10	20	90	0	0	180	60
17	5	90	25	160	90	10	175	90	90	60	180	60
18	180	90	150	55	90	10	15	90	120	135	180	60
19	85	90	85	115	90	10	160	90	0	15	180	60
20	130	90	15	145	90	10	30	90	160	75	180	60