A kékkel jelölt részek csak magyarázatként szerepelnek a sablonban. Azokat mindenképpen törölni, vagy módosítani kell az alábbiaknak megfelelően!!!

**A szakdolgozatban szereplő minden szöveg feketével írandó!!!**

<A relációs jelek között szereplő szöveg helyére mindenkinek a saját magára vonatkozó adatokat kell beírni. A relációs jeleket törölni kell, csak a köztük lévő szöveg maradjon!>

{A kapcsos zárójelek közötti szöveg szükség esetén használandó. A kapcsos zárójeleket törölni kell!}

**Külső, fekete borítólap formátuma**

A pdf-ből törlendő!

|  |
| --- |
| Dolgozat gerincén, a fekete kötésen: <Dolgozat száma> |

**SZAKDOLGOZAT**

**<Hallgató neve>**

**<Neptun kódja>**

**Első, üres (kötésben használt vastagabb) lap**

A dolgozatból törlendő!

**DUNAÚJVÁROSI EGYETEM**



**<Szak neve (pl. MÉRNÖKINFORMATIKUS BSC)>**

SZAKDOLGOZAT

<dOLGOZAT CÍME>

Asboth Noémi

mérnökinformatikus jelölt

<Dolgozat száma (pl. A-003-INF-2024.)>

**Szakdolgozat témakiírása a Szakdolgozatos weboldalról exportált fájlból nyomtatva**

A pdf-ből törlendő!

**Konzultációs lap a Szakdolgozatos weboldalról exportált fájlból nyomtatva**

A pdf-ből törlendő!

**Bírálati lap a Szakdolgozatos weboldalról exportált fájlból nyomtatva**

A pdf-ből törlendő!

Nyilatkozat a Szakdolgozatos weboldalról exportált fájlból nyomtatva

A pdf-ből törlendő!

Kivonat

A kivonat célja, hogy a hallgató képes legyen a szakdolgozatának rövid, lakonikus összefoglalására olyan formában, hogy az mások számára érthető legyen. További cél a formai követelmények és a logikai felépítés betartása, amely alapján célravezető és egyben figyelemfelkeltő szöveg születhessen. A kivonat javasolt terjedelme fél, de maximum egy A4-es oldal, ahol az alcímek, irodalmi hivatkozások és lábjegyzetek tiltottak.

A kivonat eleje tartalmazza a felvetéseket és a kitűzött célokat, majd annak további része ismertesse az alkalmazott módszereket a pontos részletek kihagyásával, és emelje ki az új eredményeit.

A kivonat valóban csak a szükséges és érdekes információkat tartalmazza, amely a dolgozatot jól tükrözi. Nagyon fontos, hogy valóban megfelelő hangsúlyt kapjanak az újdonságok, mint például egy új módszer kidolgozása, optimalizálása.

Abstract

A kivonat angol nyelvű fordítása.

Tartalomjegyzék

[1. Bevezetés 1](#_Toc211178388)

[1.1. A dolgozat célkitűzései 1](#_Toc211178389)

[1.2. A dolgozat felépítése 2](#_Toc211178390)

[2. Célok és alkalmazott kritériumok ismertetése 3](#_Toc211178391)

[2.1. célok ismertetése 3](#_Toc211178392)

[2.2. wpf platform bemutatása 3](#_Toc211178393)

[2.3. asztali alkalmazás éswindows operácios rendszer választása 4](#_Toc211178394)

[2.4. csv fájlok használatának bemutatása 4](#_Toc211178395)

[2.5. fejlesztöi környezet bemutatása (wisualstudio) 4](#_Toc211178396)

[3. A fejlesztés lépéseinek bemutatása 5](#_Toc211178397)

[3.1. fejlesztési fojamat ismerttése 5](#_Toc211178398)

[3.2. UML 5](#_Toc211178399)

[3.3. GUI 5](#_Toc211178400)

[3.4. gépigény meghatározása 5](#_Toc211178401)

[4. gépészeti alapok 6](#_Toc211178402)

[4.1. a logika és a müvelet magyarázata 6](#_Toc211178403)

[4.2. képletek 6](#_Toc211178404)

[5. program bemutatása 7](#_Toc211178405)

[5.1. objektumok mükodése 7](#_Toc211178406)

[5.2. felhasználási lehetöségek 7](#_Toc211178407)

[5.3. kézikönyv 7](#_Toc211178408)

[6. tesztelés fojamatának ismertetése 8](#_Toc211178409)

[6.1. nunit tesztek 8](#_Toc211178410)

[6.2. manual teszetk 8](#_Toc211178411)

[6.3. külsö leendö felhasználok által végzett proba üzem 8](#_Toc211178412)

[7. öszefoglalás 9](#_Toc211178413)

[7.1. fejlesztési lehetöségek 9](#_Toc211178414)

[7.2. öszegzés 10](#_Toc211178415)

[7.3. Hivatkozások szabályai 10](#_Toc211178416)

[7.3.1. Az irodalmi hivatkozás szabályai a szövegben 10](#_Toc211178417)

[7.3.2. Az ábrák hivatkozásának követelményei 10](#_Toc211178418)

[7.3.3. A táblázatok hivatkozásának követelményei 10](#_Toc211178419)

[7.3.4. Mellékletek hivatkozásának követelményei 10](#_Toc211178420)

[7.4. A forráskódok elhelyezése a szövegben 10](#_Toc211178421)

[8. Összegzés 10](#_Toc211178422)

[Irodalomjegyzék 10](#_Toc211178423)

[Ábrajegyzék 12](#_Toc211178424)

[Táblázatjegyzék 12](#_Toc211178425)

[Mellékletek jegyzéke 12](#_Toc211178426)

[1. melléklet: Harci ideggázok hatásfoka 12](#_Toc211178427)

# Bevezetés

Gépészek munkája rengeteg számításból áll, ezek közül egy a forgácsolások ráhagyásának kiszámolása. Ezekhez több táblázat összevetése és képletek felhasználása is szükséges. Ezt kézzel rendkívül bonyolult és időigényes megtenni, főleg ha esetleg útközben módosítani kell a paramétereket mivel változik a megrendelő igénye vagy a gépek rendelkezésre állása ezért gondoltam hogy ezt a program ezt a szoftvert fejlesztem ráhagyás számítás az iparban rendkívül fontos hiszen ennek elmulasztása vagy hibás kiszámítása a selejtképződést eredményez

A program célja hogy egy egyszerű és átlátható szoftvert készíteni a forgácsolási megmunkálásokhoz hoz való ráhagyás kiszámítására ennek pdf-k vagy latex file-be való importálására amely könnyen olvasható és egyszerűen kezelhető ezzel megkönyitve a gépészek munkásjat.

Fontos megértenünk hogy mit is jelent pontosan a ráhagyás számítás természetesen ebben majd a későbbi fejezetekben is kitérek de összefoglalás összefoglalom itt is ráhagyás azon szám amelyet a gépészeknek rá kell hagyni úgymond az anyagra ahhoz hogy a gyártás precíz és a szabványoknak megfelelő legyen tehát egyszerűbben fogalmazva ennyivel kell nagyobb anyagot vásárolni ahhoz hogy a termék tökéletesen és precízen le gyártható legyen természetesen minimális selejt képződésével

## A dolgozat célkitűzései

**1.**A dolgozat célkitűzéseinek, céljainak és alkalmazott kritériumainak ismertetése.

**2.**A fejlesztés lépéseinek bemutatása a konzultációtól az UML- és GUI-tervezésen át a gépigény és felhasználói követelmények meghatározásáig.

**3.**A programhoz kapcsolódó gépészeti alapok, táblázatok, képletek és definíciók összefoglalása.

**4.**Az elkészült program objektumainak, felhasználási lehetőségeinek és üzleti logikájának ismertetése.

**5.**A program validálása NUnit, külső és kézi tesztelési módszerekkel.

**6.**A fő célok összegzése és a további fejlesztési lehetőségek (öntés, kovácsolás) bemutatása.

## A dolgozat felépítése

2. fejezet célokat és választási indokokat ismertet;

3. fejezet a fejlesztés lépéseit és a műszaki dokumentációt részletezi;

4. fejezet a gépészeti alapokat és képleteket rögzíti;

5. fejezet a kész program működését mutatja be;

6. fejezet a tesztelést tárgyalja; a

7. fejezet pedig összegzi az eredményeket és a továbblépési lehetőségeket.

Ez az alfejezet foglalja össze a dolgozat felépítését:

1. Az első részben, a 2. fejezetben…
2. A 3. fejezetben…
3. A 4. fejezet…
4. Az 5. fejezetben…
5. A 6. fejezetben…
6. Az utolsó részben, a 7. és 8. fejezetben…

# Célok és alkalmazott kritériumok ismertetése

## célok ismertetése

Tapasztalataim alapján a következő elvárásokat állítottam fel a saját projektemmel szemben

* **Funkcionális célok**:
  + Hossz- és felületi érdeség általános tűréseinek kezelése;
  + Ráhagyások számítása művelet- és anyagfüggő szabályok alkalmazásával;
  + Táblázatos megjelenítés és szűrés szükség esetén modosithatosága;
  + Releváns fájlformátumokkal dolgozzon.Az adatok inportálasa és exportálasa (CSV/JSON) filok bol;
  + Hibakezelés és validáció kérése a felhasználotol a fontos és nem viszavonhato müveleteknél.
  + érthetö és biztos viszajeézés az elvégzett müveletekröl.
* **Minőségi célok**:
  + UI-reszponzivitás,
  + pontosság és determinisztikus számítások,
  + tesztelhetőség (NUnit),
  + hibák visszajelzése vizuális jelöléssel,
* **Üzemeltetési célok**:
  + könnyű telepítés (MSIX),
  + kis erőforrásigény
  + offline működés.

A projekt célja egy olyan alkalmazás megvalósítása, amely a gépészeti tervezésben és gyártástervezésben gyakran előforduló feladatot – a ráhagyások és általános tűrések kezelését, számítását és megjelenítését – támogatja. Az alkalmazás Windows Presentation Foundation (WPF) technológiát használ, a fejlesztés Visual Studio környezetben történt, az adatok be- és kimenete elsődlegesen CSV/JSON formátumokra épül. Az eszköz kiemelt értéke, hogy a gépészmérnöki gyakorlatban előforduló (i) paraméteres bevitel, (ii) táblázatos összehasonlítás, (iii) számszerű validáció, (iv) kiszámított eredmények több nézetben történő megjelenítése és (v) offline működés egyaránt való támogatása. A dokumentáció célja az architektúra, a fejlesztési folyamat, a mechanikai háttér és a tesztelési módszertan egységes, konzisztens bemutatása.

A projekt fókusza: a gépészeti ráhagyás- és tűréskezelés támogatása olyan asztali szoftverrel, amely gyors, megbízható és a gyártási környezethez illeszkedik. A felhasználó elsődleges célja a pontos, ellenőrizhető eredmény; a szoftver-cél a nyomon követhetőség, a tesztelhetőség és a kezelhetőség (usability).

## wpf platform bemutatása

A Windows Presentation Foundation egy grafikus rendszer, amelyet a Microsoft fejlesztett ki felhasználói felületek tervezésére Windows alkalmazásokban. Először a .NET Framework 3.0-val mutatták be. A WPF egységes módszert kínál az alkalmazások fejlesztésére azáltal, hogy a felhasználói felületet elkülöníti az üzleti logikától. A WPF az XAML nevű nyelvet használja , amely az eXtensible Application Markup Language rövidítése, és a felhasználói felület elemeinek definiálására szolgál.[1]

A WPF (Windows Presentation Foundation) .NET-alapú UI-keretrendszer, amely deklaratív XAML-lel, erős adatkötéssel (Data Binding) és stílus/templating mechanizmussal támogatja a komplex asztali alkalmazás-fejlesztést. A projekt MVVM mintát követ: **Model** (adat és szabály), **ViewModel** (állapot és vezérlők), **View** (XAML nézetek). A WPF adatkötés és az INotifyPropertyChanged/ObservableCollection<T> használatával a számítások és listanézetek valós időben frissülnek. A hibakezelés ValidationRule/INotifyDataErrorInfo segítségével történik; formázáshoz IValueConverter osztályokat alkalmazunk. A WPF választása a táblázatos adatok (DataGrid) és a testre szabható vizuális visszajelzés miatt indokolt

## asztali alkalmazás éswindows operácios rendszer választása

A célfelhasználók tipikusan Windows-alapú munkaállomásokat használnak, nagy kijelzővel és vállalati irányítási elvárásokkal (telepíthetőség, stabilitás, offline-használat). A desktop kliens előnye a kis késleltetés, a stabil teljesítmény, a helyi fájl- és eszköz-hozzáférés. A projekt MSIX csomagolással terjeszthető, így a frissítések automatizálhatók, az eltávolítás tiszta.[https://gs.statcounter.com/os-market-share/desktop/europe/#monthly-202409-202509]

A célfelhasználók mérnöki/gyártási munkaállomásokon, jellemzően Windows alatt dolgoznak, nagy (≥22″) kijelzőkkel, sokszor többmonitoros beállításokkal. A HCI-irodalom és ipari megfigyelések egyaránt jelzik: a felhasználók a fontosabb, összetettebb feladatokat nagy kijelzőn (desktop/laptop) végzik hatékonyabban, mint mobilon. Ez a projekt paraméterezési, táblázatos összehasonlítási és validációs igényeihez illeszkedik.

A nagy/kiterjesztett kijelzők produktivitási előnyeit laborvizsgálatok is alátámasztják (térbeli emlékezet, párhuzamos kontextusok), ami a jelen alkalmazás többpaneles, adatigényes használatához különösen releváns.[ https://rauterberg.employee.id.tue.nl/conferences/INTERACT2003/INTERACT2003-p9.pdf?utm\_source=chatgpt.com]

A szoftver internet nélkül is teljes funkcionalitással használható (helyi CSV/JSON). A lokális futás determinisztikus, hálózati ingadozástól mentes; ez gyártócsarnoki környezetben kritikus.[ <https://learn.microsoft.com/en-us/dotnet/desktop/wpf/data/?utm_source=chatgpt.com>]

A HCI alapművek is hangsúlyozzák: a feladat–eszköz–környezet illesztése javítja a hibaarányt és a végrehajtási időt. learn.microsoft.com +1 2.3.8. TCO, kockázat és bővíthetőség Az asztali kliens felhő-infrastruktúra nélkül is működik, így az üzemeltetési költség (TCO) alacsony; a fő kockázat a platformhoz kötődés. Ezt rétegzett architektúrával mérsékeljük (független számítási mag + WPF UI), így jövőben alternatív felület (pl. web/MAUI) is építhető ugyanarra a magra. A döntést HCI és platform-dokumentációs források támasztják alá.

A Windows-asztali megoldás a tényleges használói környezethez és a feladat komplexitásához illeszkedik: nagy kijelzőn végzett, adatigényes, precíz interakciókra optimalizált, alacsony késleltetésű és offline-első működést biztosít. A WPF/.NET technológiai bázis és a MSIX-életciklus támogatás együttesen kockázatcsökkentő és üzemeltetés-barát választássá teszik.[ learn.microsoft.com ]

## csv fájlok használatának bemutatása

A CSV az egyik legnépszerűbb formátum az adatok webes közzétételére. Tömör, könnyen érthető mind az emberek, mind a számítógépek számára, és jól illeszkedik a legtöbb adat táblázatos jellegéhez.[ https://www.w3.org/TR/tabular-data-primer/?utm\_source=chatgpt.com]

A CSV (Comma-Separated Values) egyszerű, ember által olvasható, széles körben támogatott formátum. Előnye az interoperabilitás (Excel/Sheets kompatibilitás), a gyors feldolgozás és a verziókezelés-barát jelleg. A projektben a tűréstáblák és mintabemenetek CSV formátumúak; a belső számításokhoz szükséges komplexebb struktúrák JSON-ban is tárolhatók. A CSV beolvasás stream-alapú, nagy fájloknál lapozást/virtuális listát alkalmazunk.

## fejlesztöi környezet bemutatása (wisualstudio)

A fejlesztés Visual Studio (2022+) környezetben történt. A projekt .NET 6/7 (vagy Windows Desktop) célkeretrendszerre fordul. A Visual Studio szolgáltatásai: XAML-designer, beépített diagnosztika, NuGet-kezelés, NUnit-integráció, MSIX csomagolás. Az automatizált build folyamat a beállított konfigurációkon (Debug, Release) reprodukálható.

# A fejlesztés lépéseinek bemutatása

## fejlesztési fojamat ismerttése

A projekt inkrementális iterációkkal készült (agilis szemlélet, rövid sprintek):

Sommerville és Pressman–Maxim által is javasolt „korai és folyamatos verifikáció” elvét követi: minden iteráció végén futnak az automatizált tesztek,

1. **Követelményfeldolgozás**: use-case-k, funkcionális és nem-funkcionális követelmények rögzítése.
2. **Architektúra-terv**: rétegzett felépítés (Model–Services–ViewModel–View), adatséma és konverziók kijelölése.
3. **Prototípus**: alapszámítás és DataGrid megjelenítés; CSV/JSON beolvasás.
4. **Funkcióbővítés**: validációk, hibajelzés, export, felület finomítás.
5. **Tesztelés**: NUnit egységtesztek + kézi tesztek; edge-case-ek lefedése.

a tervezést természetesen azaá leztem hogy röviden átekintetem mimodon fogja a felhasználo igényme venni a programot

felhasználoi itipus 1. az iskolás vagy laikus felhasználo aki az oktatásban használt és a programba elöre felvitt türéseket fogja használni és részletesen átnéz minden számitást hisz tanulni szeretne belöle

***iderakok képet***

felhasználoi tipus 2. a dolgozo métnök aki valoszinüleg a saját cége által használt pontosabb türési kritériumokal dolgozn a igy lehetöséget adok számára a táblák szerkesztésére ere az automatikus exeles megnyitást választotam abbol az okbol amiért ez elég közismert és az idösebb mérnökök is könyedén igazodnak el emellet természetesen a csv filok kézi szerkesztése is lehetöség.

***ide is kép kell***

## UML

## GUI

Validáció vizuális jelzésekkel, ToolTip-ekkel, állapotsorban összefoglaló üzenetekkel.

a grefikus felület megtervezésénél figyelembevettem hogy a gépészek által gyakran használt programok ne a modern programok kinézeti mintáira hajaznak hanem sokal inkább a 2000 évek elejének egyszerüb de pixelesebb grefikájat részesitik elönyben föként mivell ekor készültek ezek a programok igy én probáltam egy az ehez szokott felhasználok számára is érthetö mégis letisztultabb és egyértermübb kezelö felületet létrehozni. a szinválasztásikal igyekeztem a nyugott és koncentrált munkamenetet támogatni igy a fö szinek a fehér és a kék különbözö lágy árnyalatai lettek ezel szemben a fontos és kritikus megerösitéseknél figyelemfelhivo szineket használok. a korábban a célok közt emlitett fontos kis gép igény miatt hisz a diákot lehetséges hogy nem rendelkeznek tul jo pc vel és emellet az aplikácio mellet várhato a más appok futataása igy jo ha ez a program minél kevesebb eröforrást igényel igy az animáciokat és a nagy felbontást kerültem

## gépigény meghatározása

* **Minimális**: 2 magos CPU, 4 GB RAM, 200 MB tárhely, Windows 10/11, .NET Desktop Runtime.
* **Ajánlott**: 4 magos CPU, 8–16 GB RAM, SSD, FHD kijelző, Windows 11.
* A futás CPU- és memóriaigénye lineárisan nő a betöltött adatsorok számával; streamelt beolvasás és lusta materializáció csökkenti a csúcsterhelést.

# gépészeti alapok

## a logika és a müvelet magyarázata

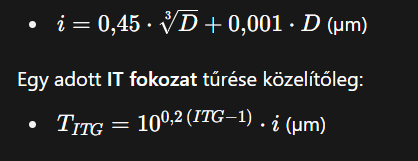
Ennek ellenére eldöntheti, hogy egy gyártott termék mennyire térhet el a műszaki rajz méreteitől, és még mindig elfogadható legyen. A gyártásban ezt az elfogadási tartományt tűréshatárok határozzák meg. Ezek a tűréshatárok a névleges méretek (a terv eredeti szándéka) és a méret maximális és minimális értékei közötti eltéréseket jelentik, amelyek még garantálják az illeszkedést. Ezeket a gyártási tűréseket szabályozott hibahatárnak is tekinthetjük. [https://www.fictiv.com/articles/iso-2768-an-international-standard?utm\_source=chatgpt.com]

**Ráhagyás**: az az anyagfelesleg, amelyet a megmunkálás során eltávolítunk a kívánt készméret elérése érdekében. A ráhagyás számításnál a technológiai szórás, az érdességi követelmény és a gép–szerszám rendszer adottságai figyelembe veendők.

**Általános tűrések**: ha a rajzon nincs külön tűrés megadva, a vonatkozó szabvány szerinti általános tűrések érvényesek (lineáris és szögméretekre). A szoftver a CSV/JSON táblából olvassa be a tartomány–IT osztály–tűrés kapcsolatait. A „kész” méret(ek) és az „előgyártmány” méret(ek) közötti különbséget a **ráhagyás** adja, amely az alkalmazott megmunkálástól (esztergálás, marás stb.), az anyagminőségtől és az előkészítéstől függ.

## képletek

A hosszméretek általános tűrései gyakran az ISO 286 sorozatra és az ISO 2768 általános tűrésekre vezethetők vissza. A **tűrési egység** (i) közelítése (μm-ben) az alábbi empirikus összefüggéssel adható meg (D [mm] a névleges méret):



Az ISO 2768-1 általános tűrései (lineáris és szögméretek) több osztályba sorolják a megengedett eltérést a névleges mérettartomány függvényében. A szoftver ezekből a táblákból választ a méret és az osztály alapján. ***(****A szabványok szerinti pontos értékeket és táblákat a felhasználó szervezeti hozzáféréssel, vagy gyártási előírások alapján adják meg; a szoftverparaméterezés ezt kiszolgálja.)*

# program bemutatása

## objektumok mükodése

 **ToleranceEntry**: egy tűrésrekord; mezők: RangeMin, RangeMax, Kind (Length/Angle), IT, ToleranceValue.

 **AllowanceRule**: ráhagyás-szabály; mezők: Operation, Material, BaseAllowance, Corrections.

 **ToleranceProvider**: CSV/JSON beolvasás, séma-ellenőrzés, hibatűrő parse (idézőjelek, elválasztók, kulturális formátumok kezelése).

 **AllowanceCalculator**: képletek, IT→T konverzió, tartomány-keresés, edge-case kezelés.

 **MainViewModel**: állapot (ObservableCollection<T>), parancsok (ICommand), validáció (INotifyDataErrorInfo).

 **ExportService**: CSV/JSON export, pontos kerekítés és lokalizáció.

## felhasználási lehetöségek

## kézikönyv

# tesztelés fojamatának ismertetése

nem megfelelő tesztelés” következményei négy csatornán jelennek meg: (1) látványos hibák és meghibásodások; (2) magasabb fejlesztési költségek (sok későn talált hiba drágán javítható); (3) hosszabb piacra jutási idő; (4) magasabb piaci tranzakciós költségek (pl. óvatosságból redundáns rendszerek fenntartása, több beszállítói egyeztetés). egységes fogalmak, formátumok, interfészek és mértékegységek nélkül a teszteredmények nem hordozhatók, az összehasonlíthatóság sérül, és az automatizálás megtorpan. egységes fogalmak, formátumok, interfészek és mértékegységek nélkül a teszteredmények nem hordozhatók, az összehasonlíthatóság sérül, és az automatizálás megtorpan.[ <https://www.nist.gov/system/files/documents/director/planning/report02-3.pdf>][https://cdn.standards.iteh.ai/samples/35733/2ca18b477b7845a5b8cae39d6de0c098/ISO-IEC-25010-2011.pdf?utm\_source=chatgpt.com]

## nunit tesztek

A Microsoft ajánlása a unit tesztekre: legyenek kicsik, izoláltak, AAA mintával szervezettek;[ <https://learn.microsoft.com/en-us/dotnet/core/testing/unit-testing-best-practices?utm_source=chatgpt.com>]

Kövesd az AAA mintát, tartsd kicsiben és izoláltan; drótokra/IO-ra ne támaszkodjon; mérd és értelmezd a kódfedettséget (de ne hajszold öncélúan a 100%-ot). Üzemeljenek a CI-ben minden pushra.[ https://learn.microsoft.com/esn-us/dotnet/core/testing/unit-testing-best-practices?utm\_source=chatgpt.com]

ciklomatikus komplexitás:

A képletek és szabályok egységtesztjei a következőket fedik le: (i) IT→T konverzió per mérettartomány, (ii) ráhagyás számítás tipikus és szélső esetekben, (iii) hibás inputok (negatív méret, ismeretlen IT), (iv) CSV/JSON parse hibák. A tesztek izoláltak, mock-olt adatforrásokkal futnak. A build pipeline része a tesztfuttatás és a lefedettség-jelentés.

Az assertionök (állítások) központi szerepet játszanak az xUnit keretrendszerek egységteszteléseiben, és az NUnit sem kivétel. Az NUnit az Assert osztály statikus metódusaiként gazdag assertion készletet biztosít.

Ha egy állítás sikertelen, a metódushívás nem ad vissza értéket, és hibát jelez. Ha egy teszt több állítást tartalmaz, akkor a sikertelen állítást követők nem kerülnek végrehajtásra. Emiatt általában a legjobb, ha tesztenként egy állítást próbálunk meg végrehajtani.

Minden metódus meghívható üzenet nélkül, egyszerű szöveges üzenettel, vagy interpolált argumentumokat (más néven interpolált karakterláncot) tartalmazó üzenettel.

Assert.ThatAz NUnit 3.0-s és újabb verzióiban az állítások elsősorban a metódus használatával íródnak , amely korlátozó objektumokat fogad argumentumként. Ezt az állítások korlátozó modelljének nevezzük.

Unit (komponens) teszt: kis, elszigetelt kódrészek (függvények/osztályok) automatizált vizsgálata, gyakran stubokkal/driverekkel. A nemzetközi szakirodalom a „component (unit) testing”-et önálló komponensekre fókuszáló tesztszintként definiálja.

erikvanveenendaal.nl.

## manual teszetk

## külsö leendö felhasználok által végzett proba üzem

Pilot üzem valós adatokkal: feladatlista (use-case-ek), visszajelző űrlap, hibajegy-kezelés. A tanulságok alapján UX finomítás, dokumentáció pontosítás, alapértelmezések beállítása.

# öszefoglalás

A projekt demonstrálja, hogy a WPF-alapú, Windows asztali alkalmazás alkalmas a gépészeti tűrés- és ráhagyásszámítás hatékony támogatására. A technológiai választások (WPF/MVVM, CSV/JSON, MSIX) a célterület igényeire szabottak: precíz vezérlők, táblázatos megjelenítés, alacsony késleltetés, offline működés, egyszerű telepítés. A tesztelési stratégia és a dokumentáció biztosítja a minőségi követelmények teljesülését.

## fejlesztési lehetöségek

* Teljes körű szabványtáblák integrációja (licencelt forrásból), verziókövetéssel.
* Szenzor-/perifériaintegráció (soros port, mérőeszközök), kalibrációs modul.
* Felhős szinkron (opcionális), jogkör-kezelés, napló-központ.
* Teljesítmény-optimalizálás (virtualizáció javítása, async I/O finomítása).
* öntési és kovácsolási ráhagyás felvétele

## öszegzés

Minden Címsor1 stílusú sor új oldalra kerüljön.

A címsorokat soha ne szerepeltessük úgy, hogy azt közvetlenül egy új, alacsonyabb szintű címsor követi. Minden esetben írjunk alá legalább 2-3 sor terjedelmű szöveget. Továbbá, abban az esetben, ha a fejezetnek csak egy alfejezete van, a fejezet alfejezetre bontása felesleges. Ilyenkor a fejezeten belül kell a mondanivalót leírni.

## Hivatkozások szabályai

A következő alfejezetekben áttekintjük a különböző szakdolgozati elemek hivatkozásának szabályait.

### Az irodalmi hivatkozás szabályai a szövegben

Amennyiben egy darab irodalomból szeretnénk nem szószerinti hivatkozást végrehajtani egy mondat esetében azt a mondat végi írásjel előtt tegyük meg, ahogy az alábbi példa is mutatja:

Ez egy példa mondat a fentebb tárgyalt hivatkozási típus bemutatására [1].

Ha egy teljes bekezdést hivatkozunk egy darab irodalomból, akkor a hivatkozás számát a bekezdés végére tegyük, a mondatvégi írásjel után.

Ezek egy példa bekezdés mondatai a fentebb tárgyalt hivatkozás típus bemutatására. Ezek egy példa bekezdés mondatai a fentebb tárgyalt hivatkozás típus bemutatására. Ezek egy példa bekezdés mondatai a fentebb tárgyalt hivatkozás típus bemutatására. [1]

Ha több irodalomból hajtunk végre hivatkozás jelölést, akkor azt az irodalomjegyzékben nem a sorszámban egymást követő irodalmak esetében vesszővel soroljuk fel, ha sorszámban egymást követik, akkor kötőjellel szerepeltessük. Az alábbi példa egy mondat esetében mutatja a megfelelő hivatkozási formát:

Ez egy példa mondat a fentebb tárgyalt hivatkozás típus bemutatására [1-4], [5], [10].

Ha egy teljes bekezdést hivatkozunk több irodalomból, akkor a hivatkozás számát a bekezdés végére tegyük, a mondatvégi írásjel után.

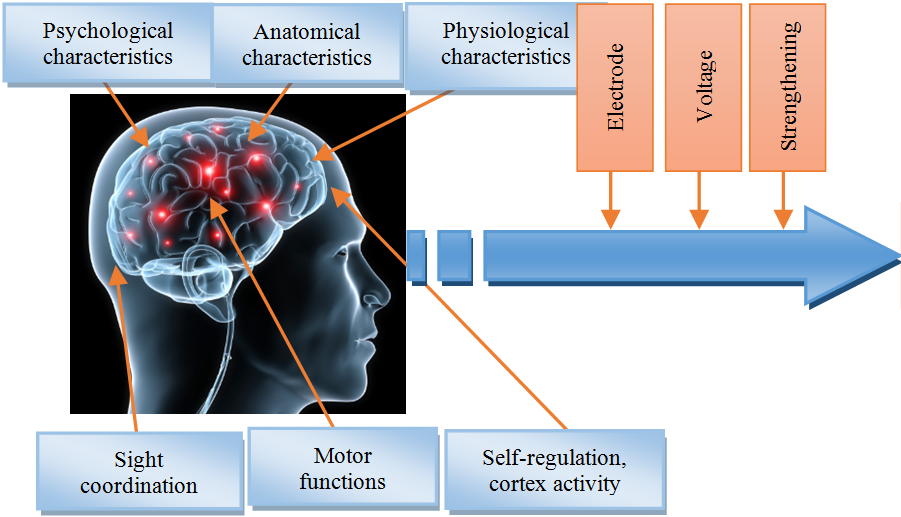
Ezek egy példa bekezdés mondatai a fentebb tárgyalt hivatkozás típus bemutatására. Ezek egy példa bekezdés mondatai a fentebb tárgyalt hivatkozás típus bemutatására. Ezek egy példa bekezdés mondatai a fentebb tárgyalt hivatkozás típus bemutatására. [1-4], [5], [10]

A szószerinti idézetet minden esetben idézőjelek közé tegyük, jobbra igazítsuk és dőlt betűformázással jelöljük, továbbá az idézet végére tegyük a hivatkozást, majd új bekezdésben folytassuk a dolgozatot.

*„*Ez egy példa a szószerinti hivatkozás jelölésére*.” [1]*

### Az ábrák hivatkozásának követelményei

Az ábrákat az 1. ábrának megfelelően helyezzük el a dolgozatban, azaz középre igazítva, előtte 18pt nagyságú térközt, valamint 1,5 sor nagyságú sorközt hagyva, az utána 18pt nagyságú térközt a sablon megoldja, illetve a könnyebb olvashatóság végett megfelelő felbontással és nagyságban. Az ábráknak kell lenni számának és címének, amire a szövegben minden esetben hivatkozzon! A képaláíráshoz lehetőség szerint a Word szolgáltatásai között megtalálható felirat beszúrását használja. Amennyiben saját szerkesztésű ábrát használ fel, akkor az irodalom hivatkozás helyére az alábbi szöveget írja: (saját szerkesztésű ábra).



1. ábra: Bioelektromos jelek feldolgozásának egyszerű modellje [1]

Az ábrákat a megelőző, illetve következő oldalra is beszúrhatjuk, elkerülve ezzel az oldal alján az üres részeket.

### A táblázatok hivatkozásának követelményei

Ahogy az ábrákra, úgy a táblázatokra is hivatkozni kell a szövegben, például: Az 1. táblázat mutatja az eredményeket. Ehhez a táblázatok számát és nevét lehetőség szerint a Word szolgáltatásai között megtalálható felirat beszúrásával oldja meg! Ahogy az ábrák esetében, a táblázatokat is igazítsa középre, előtte és utána 1-1 normál stílusú üres sort hagyjon, valamint a könnyebb olvashatóság végett megfelelő színekkel kombinálva tegye. Az 1. táblázat stílusát nem kötelező követni, de ajánlott. Ha a táblázatot épp megtörné például egy oldal, akkor ne hagyjuk „félbevágva”, alkalmazzunk oldaltörést.

1. táblázat: Eredmények [1]

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Tantárgy** | **Csoportok** | **Átlag** | **Szórás** | **Hatásnagyság** | **t-próba** |
| Matematika | G1 | 3,036 | 0,769 | 0,14\* | t = 0,53  (p < 0,05) |
| G2 | 3,15 | 0,933 |
| Elektronika | G1 | 3,345 | 0,886 | 0,05\* | t = 0,19  (p < 0,05) |
| G2 | 3,3 | 0,979 |
| Programozás | G1 | 3,254 | 0,927 | 0,05\* | t = 0,18  (p < 0,05) |
| G2 | 3,3 | 0,979 |

\**kicsi hatásnagyság (d<0.5).*

A táblázatokat a megelőző, illetve következő oldalra is beszúrhatjuk, elkerülve ezzel az oldal alján az üres részeket.

### Mellékletek hivatkozásának követelményei

Ahogy az ábrákra, táblázatokra, úgy a mellékletekre is hivatkozni kell a szövegben, például: Az 1. számú mellékletben található módon ábrázoltam a jelenlegi állapotot.

## A forráskódok elhelyezése a szövegben

A forráskódok beillesztésére soha ne alkalmazzunk képeket, azokat szövegként jelenítsük meg a dolgozatban. Lehetőség szerint a törzsszövegtől eltérő betűtípust alkalmazzunk, például Consolas betűtípust. A több oldalas kódokat ne a szövegtörzsbe, hanem a mellékletekbe rakják és itt szorítkozzunk csak a lényeges elemek kiemelésére. A legelegánsabb, ha valamilyen nyelvfüggetlen leírást használ az algoritmus bemutatásához és nem egy az egyben emeli be az adott nyelven íródott kódbázist a dolgozatba. Alább láthat egy-egy példát:

Az alábbi nyelvfüggetlen forráskód részlet jelentős részben hozzájárult az xy algoritmus optimalizálásában:

Algorithm arrayMax(A,n)

currentMax ← A[0]

for i ← 1 to n-1 do

if A[i] > currentMax then

currentMax ← A[i]

{ increment counter i }

return currentMax

Az alábbi C# nyelven íródott forráskód részlet jelentős részben hozzájárult az xy algoritmus optimalizálásában:

int? currentMax = null;

int index = -1;

for (int i = 0; i < array.Length; i++)

{

int currentNum = array[i];

if (!currentMax.HasValue || currentNum > currentMax.Value)

{

currentMax = currentNum;

index = i;

}

}

# Összegzés

Ebben a fejezetben röviden foglalja össze a dolgozat lényegét, nagy hangsúlyt fektetve a saját eredményeire. Javasolt terjedelem 1 oldal.

# Irodalomjegyzék

[1] <https://blog.postsharp.net/wpf-best-practices-2024?utm_source=chatgpt.com>

A felhasznált szakirodalmat []-ben sorszámozottan kell feltüntetni. Internetes forrás esetén a megtekintés idejét is meg kell jeleníteni. Kötelező hivatkozási stílus: IEEE, ehhez lehetőség szerint a Word szolgáltatásai között megtalálható forráskezelést használja. Alább láthat példákat különféle irodalmi típusok hivatkozására. A dolgozat legalább 15-20 db hivatkozást tartalmazzon, lehetőség szerint minden típusból néhányat.

Folyóiratcikk:

[1] J. Katona and A. Kovari, "A Brain–Computer Interface Project Applied in Computer Engineering," in *IEEE Transactions on Education*, vol. 59, no. 4, pp. 319-326, Nov. 2016.

Könyv:

[2] J. Sharp, *Microsoft Visual C# 2005 lépésről lépésre.* Szak kiadó Kft., Bicske, 2005.

Könyvrészlet:

[3] J. Katona, T. Ujbanyi, G. Sziladi, A. Kovari: *Electroencephalogram-based brain-computer interface for Internet of Robotic Things*, Cognitive Infocommunications, theory and applications, Topics in Intelligent Engineering and Informatics, Springer, pp. 249-272, 2018.

Konferenciaközlemény:

[4] J. Katona, T. Ujbanyi, G. Sziladi and A. Kovari, "Speed control of Festo Robotino mobile robot using NeuroSky MindWave EEG headset based brain-computer interface," *2016 7th IEEE International Conference on Cognitive Infocommunications (CogInfoCom)*, Wroclaw, 2016, pp. 000251-000256.

Internetes hivatkozás:

[5] *EEG Headsets | NeuroSky Store*, Store.neurosky.com, 2017. [Online], Elérhető: [https://store.neurosky.com](https://store.neurosky.com/) [Letöltve: 2017. 07. 24.]

# Ábrajegyzék

[1. ábra: Bioelektromos jelek feldolgozásának egyszerű modellje [1] 3](#_Toc8579253)

# Táblázatjegyzék

[1. táblázat: Eredmények [1] 4](#_Toc8579262)

# Mellékletek jegyzéke

1. melléklet: Harci ideggázok hatásfoka

# melléklet: Harci ideggázok hatásfoka

A dolgozat mellékletei sorszámozottan, ha vannak. A Mellékletek jegyzéke kereszthivatkozással hivatkozzon az adott mellékletre!