Tesztelés

Tartalom

[Tesztelési alapok 2](#_Toc469608928)

[Tesztelés típus (résztvevők alapján) 2](#_Toc469608929)

[Debug 2](#_Toc469608930)

[Tesztelés bemenete 3](#_Toc469608931)

[Tesztelés és a mérföldkövek 3](#_Toc469608932)

[Tesztelés típus (fókusz alapján) 3](#_Toc469608933)

[Tesztadatok 4](#_Toc469608934)

[Adattárolás 4](#_Toc469608935)

[Tesztelés típus (terület alapján) 4](#_Toc469608936)

[Grafikus felület tesztelése 5](#_Toc469608937)

[Program bemutatása 5](#_Toc469608938)



# Tesztelési alapok

A tesztelést mindig az aktuális prototípuson végezzük, a tesztelés eredményeit és a haladást jelentjük a megrendelő által kirendelt kapcsolattartónak. A vevő biztosított számunkra már használt adatokat amelyeknek tudjuk a kimenetelét, ezért a fontosabb tesztelésekhez ezeket az adatokat használjuk, ez legfőképp a program késöbbi fázisaiban jellemző. Ha tesztelés közben nem várt eredményt kapunk akkor annak utánajárunk és kijavítjuk.

# Tesztelés típus (résztvevők alapján)

A tesztelést két nagyobb részre oszthatjuk. Az egyik tesztelés típus az amikor a megrendelő kapcsolattartója is jelen van. Ez a típus kéthetente fordul elő illetve egy komplex minden területre kiterjedő tesztelést jelent. Ez a tesztelés fajta a megrendelő tájékoztatását helyezi a közzéppontba. A másik típus arra szolgál hogy folyamatosan képet kapjunk az aktuális hibákról ez a fajta tesztelés minden módosítás után futtatandó. A tesztelések során figyelni kell az aktuális program optimalizáltságára, mivel a megrendelő legfőbb kérése az volt, hogy a lehető legkevesebb erőforrás felhasználásával a lehető leggyorsabban számítsa ki a feladat az eredményt.

# Debug

Az esetleges felmerülő problémák elhárítására debug-olást alkalmazunk ennek segítségével lépésről lépésre megvizsgáljuk a program futását. A jelentkező problémákat a lehető leggyorsabban megpróbáljuk orvosolni. Amennyiben kijavítunk egy hibát, újra tesztelést kell futtani, hogy a javítás nem eredményezett-e egy újabb pontatlanságot.

# Tesztelés bemenete

A teszteléseket két féle bemenettel végeztük. Az aktuális prototípus lefuttatásra kerül a megrendelő álltal szolgáltatott adatokkal illetve random generált számokkal. A kapott eredmény minden esetben ellenőrzésen esik át. Amennyiben különbség jelentkezik a számított adat és a várt eredmény között nyers erős módszerrel leellenőrizzük, hogy melyik számítás a téves. Amennyiben a számított adat hibás abban az esetben a programon újabb hibakeresést végzünk el.

A tesztelések során megpróbáljuk létrehozni azt a végleges prototípust amelyet végül a megrendelő számára használatra bocsájtjuk.

# Tesztelés és a mérföldkövek

A program tervezése során meghatároztunk különböző mérföldköveket amelyek elérése esetén kötelező tesztelést végezni. A mérföldkövek előrehaladtával egyre több mindenre kell figyelni, hogy a javítással ne válltoztassuk a program többi jól működő részét negatív irányba.

# Tesztelés típus (fókusz alapján)

A tesztelés fokuszától függően két részre oszthatjuk. Kisebb részeket vizsgálunk elszeperáltan a többitől. Ilyenkor csak az adott kódrészre koncentrálunk, hogy a figyelem csak a program egy kisebb részére öszpotasítson így esetlegesen nagyobb esély lehet a hiba detektálására. A másik ellenőrzés során a programot egészében tekintjük. Figyelünk az egyes függvények egymásra gyakorolt hatására. A tesztelés során először a kisebb programrészek elszeparált tesztelése történik majd a megoldás megtalálása után a program egészét tekintve vizsgáljuk a megoldást. Amennyiben megoldást találtunk a detektált hibára, újabb tesztelést futtatunk a már elkészített programrészre. Amennyiben a program a megfelelő kimenetet generálja továbblépünk a program fejlesztésében és a következő felmerülő problémára keressük meg a megoldást.

# Tesztadatok

A tesztelést több bemenetre is elvégezzük. A többféle adatra történő tesztelés arra szolgál, hogy ne forduljon elő az, hogy véletlenül generálódjon jó eredmény és ez miatt a késöbbiekben legyen egy komolyabb hiba, amely jelentősen megnövelheti a fejlesztési időszak időtartamát, költségét.

# Adattárolás

Kiemelt figyelmet kell fordítani az adat ideiglenes tárolására a tesztelés során. Amennyiben a bevitt adatok hibásan kerülnek eltárolásra a logikai rész rossz eredményt fog számítani nagy valószínűséggel.

# Tesztelés típus (terület alapján)

Az tesztelést terület alapján három részre oszthatjuk:

* Adattárolás: Az adatok megfelelő eltárolásának vizsgálata.
* Adatbekérés: A grafikus felületért felelős programkód ellenőrzése.
* Logikai rész: A számításokért felelős programkőd tesztelése

# Grafikus felület tesztelése

A grafikus és beviteli felület elkészülte után elősször egyszerű kiíratással NetBeansben teszteljük a bevitelt, ezt nagymennyiségű adattal tesszük hogy semmiképp ne keletkezzen hiba. Ha ez sikeres akkor a megbízótól kapott adatokkal folytatjuk a működés tesztelését.A kíírási felület elkészülte után teszteljük hogy valóban a várt eredményeket kapjuk vissza, a vevő által szolgáltatott adatokból.

# Program bemutatása

A vevő által kijelölt közvetítőnek. Bemtatjuk a program egy tesztelését az kapcsolattartónak és hagyjuk, hogy kipróbállhassa. A bemutatás célja, hogy a megrendelő megfelelő képet kapjon a program aktuális állásáról. A bemutatás után a vevő elmondja a véleményét az aktuális prototípusról. Lehetőleg ezen tesztelés alatt ne jelenjen meg váratlan hiba. A tesztelés után folytatódik a további fejlesztés.