**I. Introducere. Elemente teoretice relative la tema lucrării de licență.**

***I.1. Justificarea alegerii temei (privire de perspectivă, dorința de îmbunătățire a tehnicii actuale de pescuit sportiv);***

Alegerea temei,“ Caracteristici teoretice și practice ale unor dispozitive utilizabile în pescuitul sportiv, evidențiate în mediul MATLAB”, se datorează unei pasiuni mai vechi pentru pescuit .

 Pescuitul sportiv este sportul caracterizat prin prinderea peștelui de apă dulce sau sărată cu ajutorul unui băț (varga), linie și cârlig. Asemenea vânătorii, pescuitul, își are originea în nevoia oamenilor de a obține și produce alimente pentru supraviețuire.

     Pescuitul sportiv își are originile încă din antichitate. O scenă egipteană datată aproximativ din anul 2000 i.H., înfățișează o secvență de pescuit cu undiță și plase.



O relatare din China din secolul 4î.H. se referă la pescuitul cu o linie de mătase, cârlig și băț de bambus, iar ca momeală se folosea orez fiert. Pe lângă acestea, referiri la pescuit, sunt întâlnite în limba greacă veche, asiriană, romană și în scrieri evreiești.

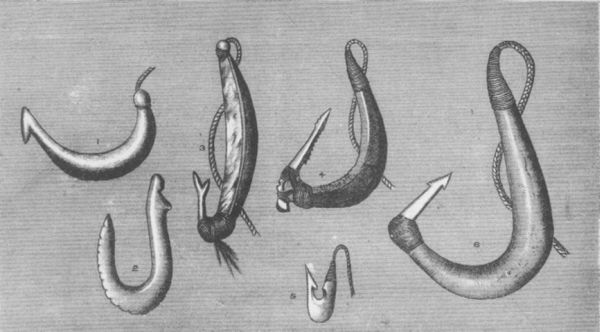
Astăzi, pescuitul, întâlnit adesea sub denumirea de pescuit sportiv, pentru a fi diferențiat de pescuitul comercial, este, în ciuda creșterii orașelor și a poluării tot mai mari, un mod de relaxare și pentru multe țări unul dintre cele mai populare sporturi. Însă problemele pescarului modern rămân în esență aceleași cu ale strămoșului său: unde poate gasi peștele, cum să se apropie de el și ce fel de momeală să folosească pentru a-l prinde.

    De asemenea pescarul trebuie să înțeleagă factorii naturali, cum ar fi vântul și vremea pentru o partida de pescuit reușită. Prin urmare pescuitul rămâne o veșnică problemă de rezolvat a istoriei naturale.

Istoria pescuitului este legată în mare parte de echipamentul(sculele) de pescuit.

Unul dintre primele articole de pescuit confecționate de om, considerat și printre primele unelte ale omului, a fost cârligul de pescuit. Strămoșul cârligului de astăzi era confecționat dintr-o bucată de lemn, os sau piatră alungite, ascuțite la ambele capete și perforate la mijloc. Acest cârlig era uns sau acoperit cu o momeală, care determina pestele să-l înghită. Când se trăgea de fir, cârligul rămânea blocat în esofagul peștelui, dând astfel posibilitatea pescarului să scoată peștele din apă.

Odată cu apariția metalelor, cârligul de pescuit a fost printre primele unelte confecționate de om. La acest cârlig s-a atașat un fir de origine animală sau vegetală, metoda fiind eficientă doar atunci când era folosită la bordul unei ambarcațiuni. Dar, mai tarziu, acestea au fost atașate la o vargă, care probabil la început a fost un băț sau o ramură, ceea ce a facut posibil pescuitul de pe mal și dincolo de vegetația învecinată acestuia. Mii de ani varga undiței a rămas foarte scurtă nedepășind în lungime câțiva metri. Cea mai timpurie referință la o vargă mai lungă, este din epoca romană. La acel moment, se relata despre pescarii macedoneni care pescuiau păstrăv la muscă artificială și a fost descris modul în care aceștia făceau acest lucru. Varga avea lungimea de 6 picioare (1,8 metri) și linia, aceeași dimensiune, ceea ce făcea ca musca să poată fi ușor așezată pe suprafața apei.



  Istoria pescuitului sportiv, a început în Anglia cu tipărirea de către Wynkyn de Worde a cărții Treatyse de Fysshynge With an Angle (1496) ca parte a celei de a doua ediții din The Böke de St Albans, care se ocupa inițial numai de vânătoare. Cartea a fost bazată pe subiecte tratate pe continent în secolul al 14-lea. Metodele de pescuit descrise în Treatyse sunt surprinzător de moderne (șase din cele douăsprezece menționate sunt încă în uz). Undițele descrise în lucrare, sunt de 10-12 metri lungime, cu fir (linie) din păr de cal împletite legat de unul din capete.

Prima perioadă de mare avânt a pescuitului sportiv, a venit pe la mijlocul secolului al 17-lea, atunci cand Izaak Walton și Charles Cotton au scris The Compleat Angler iar colonelul Venables Robert și Thomas Barker au abordat și prezentat noi metode de pescuit. În această perioadă, pescari necunoscuți, atașează la capătul bățului de pescuit o buclă de sârmă sau un inel, realizând astfel o cale de rulare pentru fir, utilă atât la aruncare cât și la a obosi peștele.

  În 1667 este menționată o linie de prins somon de 26 de metri. Acest lucru face evident faptul că era nevoie de ceva care să strângă și să elibereze firul acestei linii, fapt care ne trimite cu gândul că,în acea perioadă a fost inventat stramoșul mulinetei de astăzi, care la acea vreme arăta asemenea unui tambur pe care se strângea firul.



Prima mulinetă rudimentară a constat într-o bobină de lemn cu un inel de metal care era montat pe degetul mare al pescarului. Prin 1770 a fost montată pe vargă, de-a lungul ei,o bară cu inele pentru ghidare. Prima mulinetă adevarată a apărut atunci când pe vargă, în apropierea mânerului a fost montat un tambur cu a bobină care permitea rotirea și strângerea firului. Acest tip de mulinetă a fost cunoscut și făcut popular de doi ceasornicari din Kentucky la începutul anilor 1800.

       Odată cu apariția mulinetei, pescuitul de toate felurile a devenit mult mai ușor, mai ales cel la muscă, care acum, permitea muștelor să plutească pe apă, în lungul curentului. În timp, lansetele s-au îmbunătățit considerabil, lemnul greu fiind înlocuit cu lemn cu granulație mai dură sau cu esențe elastice cum ar fi lancewood și greenheart din America de Sud si Indiile de Vest, precum și de bambus. Până la sfârșitul secolului al 18-lea tehnicile au fost dezvoltate, astfel încât mai multe benzi de bambus și trestie de zahăr au fost lipite împreună, obținându-se forță și elasticitate, dar, în mare măsură reducerea grosimii.

   Între 1865 si 1870 vergile complet hexagonale, făcute din șase benzi triunghiulare din bambus, au fost produse de pe ambele maluri ale Atlanticului. Din 1880 metodele de fabricare a undițelor au evoluat rapid. Părul de cal pentru firul de pescuit a fost înlocuit cu mătase acoperită cu straturi de ulei obținut din semințe de in. Fiind gresate, firele nu se îmbibau cu apa, astfel, pescarul putea arunca la distanțe mult mai mari ca până acum.

În Nottingham, a apărut mulineta din ebonită (un cauciuc dur) sau metal înlocuind lemnul, astfel încât aceasta a devenit și mai ușoară la filare. În 1896 William Shakespeare, din Kalamazoo, Michigan, a conceput un tambur care aduna firul în mod uniform. În 1880 firma Malloch, din Scoția, a introdus prima bobină cu platan, care a avut o parte a bobinei deschisă. În timp, tamburul a fost întors cu 90 º, aducându-l în aceeași linie cu ghidajele de pe vargă. Mulinetele devin din ce în ce mai rezistente, ajungând să poată fi utilizate inclusiv la pescuitul la somn.

În secolul 20, vergile au devenit mai scurte și mai usoare, fără a sacrifica puterea. Bambusul a fost în mare parte înlocuit cu fibra de sticla și, în final cu fibra de carbon. După anii 1930 tamburul suveică a fost preluat în Europa iar după al doilea razboi mondial, în America de Nord și restul lumii. Linia de naylon monofilament a fost dezvoltată la sfârșitul anilor 1930 și a devenit dominantă dupa al doilea razboi mondial; dar, pe lângă ea au fost utilizate și alte împletituri din alte material sintetice. Acoperirea firului cu material plastic le-a permis să plutească sau să scufunde fără a fi nevoie de ungere.

În ultima parte a secolului 20 pescuitul sportiv a înregistrat o creștere spectaculoasă.

Odată cu dezvoltarea transporturilor, dar mai ales a celor aeriene, pescuitul în diferite zone, a devenit accesibil tot mai multor pescari. Pescuitul sportiv s-a diversificat, devenid populare metode de pescuit atât pentru apa sărată, dulce, în bălți, în ape curgătoare cât și pe mări si oceane. Au apărut recorduri din ce in ce mai mari, iar pentru diferite specii de pești au apărut metode din ce în ce mai performante de pescuit.

Cu fiecare zi ce trece, accesoriile pentru pescuit capătă o tot mai mare importanță și nu se mai poate concepe o partidă de pescuit în care să nu utilizăm câteva accesorii, altele decât pluta sau acul. De-a lungul timpului, am observat că și la pescuit, marii campioni ai lumii colaborează cu producătorii de articole de pescuit, elaborând cele mai variate și mai utile accesorii, tehnologia începând să preia controlul. Au apărut în ultimii ani mai multe aparate, cum ar fi: sonarul, navomodelul, senzorii care avertizează că peștele se află la partea cealaltă a firului și chiar swinger-ul care odată conectat la un senzor, în momentul trăsăturii, face un semnal luminos de mare ajutor pe timp de noapte. Pornind de la ceea ce există deja pe piață, m-am gândit ce anume ar mai fi util unui pescar? Astfel, am decis că un dispozitiv care să indice lungimea firului lansat, unul care să indice greutatea și poziția peștelui în apă, atunci cand ar ajunge pe piață, ar fi capabile de a genera satisfacții celor ce le vor utiliza.

***I.2. Precizarea unor dispozitive actuale de pescuit sportiv***

În momentul de față există în magazine, ca principale dispozitive utile în pescuit sonarele, navomodelele, senzorii și swinger-e .

1. **Sonarul**

Sonarele sunt dispozitive electronice prevăzute cu un senzor, denumit și traductor, de obicei dublu-cristal, care conține mai multe foițe de cristal de cuartz suprapuse. Cuartzul se bucură de proprietatea că își modifică volumul la trecerea unui curent electric prin masa acestuia și, invers, generează curent când asupra sa se exercită o presiune exterioară (efectul piezoelectric). Atunci când îl alimentăm la un curent de înaltă frecvență cristalul vibrează cu acea frecvență luând naștere o undă elastică - ultrasunetele. Undele emise prin apă lovesc obstacolele, corpuri cu densități diferite de cea a apei, fiind reflectate. Odată reflectate, o parte din acestea ajung din nou la traductor unde sunt tranformate de cristalul de cuartz (receptor) în semnal electric. Semnalul electric este prelucrat rezultând o imagine electronică, afișată pe display-ul sonarului. În funcție de complexitatea sonarului, alături de vizualizarea peștilor aflați în zona fascicolului ultrasonic mai putem obține și alte informații precum viteza peștilor aflați în mișcare, temperatura apei. Majoritatea sonarelor dispun de filtre care permit afișarea peștilor pe ecran pe game de dimensiuni. Mai facem precizarea că, consecință a unor legități ale fizicii, undele folosite aici sunt numai unde longitudinale, emise în plan normal la suprafața traductorului.

Sonarul, asemănător ca principiu de funcțonare cu GPS-ul,este un sistem care folosește proprietățile propagării sunetului în apă, pentru a detecta si localiza obiectele submersate sau pentru a măsura distanțele subacvatice. Inițial, dispozitivul a fost folosit, în timpul Primului Război Mondial, de către marina militară, pentru reperarea submarinelor și a minelor, pentru ghidajul torpilelor, pe sub apă, către ținte, ulterior utilizarea lor extinzându-se la măsurarea adâncimilor, pentru navigația maritimă și fluvială, la pescuitul comercial, la detectarea bancurilor de pești, pentru siguranța scufundărilor și pentru stabilirea comunicațiilor pe mare. Sonarul emite un impuls sonor și măsoară timpul de întoarcere al ecoului, din adâncul apei (viteza de propagare a sunetului în apa marii fiind de aproximativ 1500 m/s), date care sunt redate apoi de către operatori, prin intermediul unui monitor.

Termenul **SONAR**este acronimul de la “SOund, NAvigation and Ranging”, denumire folosită, în timpul celui de-al Doilea Razboi Mondial, de către americani. Britanicii denumesc același dispozitiv “ASDICS”, care provine de la titulatura de “Anti-Submarine Detection Investigation Committee” – “Comitetul de Investigare și detecție anti-submarină”

.

Se pare că ideea conceperii unui astfel de aparat le-a fost inspirată fizicienilor de comportamentul delfinilor, care, pentru a-și căuta hrana și pentru a comunica între ei se folosesc și ei de “sonarul” lor, amplasat în frunte. Delfinii sunt capabili sa moduleze frecvența sunetelor emise – frecvența joasă pentru detecție la mare distanță și înalta frecvență, pentru distanțe scurte.

Înainte de inventarea sonarului, la începutul secolului al XIX-lea, cercetătorul Daniel Colloden s-a folosit de un clopot subacvatic pentru a calcula viteza sunetului în lacul Geneva, din Elvetia. În 1906, constructorul de nave Lewis Nixon concepe și el un dispozitiv, similar sonarului, pentru a detecta icebergurile. Lui **Paul Langevin**, fizician francez, îi revine însă meritul de a fi realizat **primul sonar** (în 1915, la trei ani dupa naufragiul Titanicului), pentru detectarea icebergurilor și a submarinelor, denumindu-l “Ecolocație pentru detectarea aparatelor submersibile”, valorificând proprietățile piezo-electrice ale quartului (piezo-electricitatea este caracteristica unor cristale de a se polariza electric). Principiul a fost preluat de majoritatea modelelor de sonar moderne.

În momentul de față există mai multe categorii de sonare:

* **Sonarele** **active** acționează prin crearea unui puls de sunet (“șuierat”), pentru ca, mai apoi, să poată capta ecourile acestuia. În general, sonarele active, care acționează la mare distanță, folosesc frecvențe joase, cele mai joase ducând la declanșarea unui sunet de tip „BA-OANG”. Cu cât frecvența este mai scăzută, cu atât distanța de detecție este mai mare. În practică, sonarele active de foarte joasă frecvență nu coboară totuși sub 3 kHz, ceea ce înseamnă posibilitatea unor masurători pe o distanța de câteva zeci de kilometri. Pot interveni însă o serie de dificultăți, pentru că apa nu este un mediu omogen, iar propagarea sunetului poate fi perturbată de relieful subacvatic, de planton, de animalele marine etc. Alți parametri care pot influența funcționarea unui sonar sunt temperatura și presiunea apei (uneori, in mai mică măsură, și salinitatea apei).
* **Sonarele** **pasive** “ascultă”, fără a transmite sunete, aceste dispozitive fiind folosite, cu precădere, de către armată și, uneori, în scopuri de cercetare. Sonarele pasive sunt prevăzute cu impresionante baze de date sonice, din care un sistem computerizat selectează (identifică) diverse acțiuni (viteza unei nave, tipul de armament de la bord,sunetele emise de elicele vapoarelor etc.).
* Pe lângă dispozitivele active și pasive, mai există și **sonarele de** **tip “multifascicul”**, care sunt prevăzute cu un transductor extrem de performant, capabil să scaneze o arie foarte mare, la stânga și la dreapta, într-un interval de 60 de grade. Distanța și unghiul fiecărui canal pot fi folosite pentru măsurarea distanței și a profunzimii. Mai mult, cercetătorii au ajuns la concluzia că poate fi cuantificată nu doar durata ecoului, ci și amplitudinea acestuia, pentru a determina, în ultima instanță, care este gradul de reflectivitate a oceanului, dar și forma și adâncimea acestuia. ie.

Progresul tehnologic a permis dezvoltarea dispozitivelor de scanare la adancimi mari, prin detectorul de adâncime, de scanare rapidă, de scanare laterală și dispozitivul WPESS (within-pulseectronic-sector-scanning – scanare sectorială în puls). Cele mai moderne și performante sonare, dotate si cu GPS, afiseaza pe display imagini din adancul apei, peștii,  coordonatele, direcția și viteza de deplasare a celui care folosește dispozitivul. Utilizate în scopuri de cercetare, sonarele oferă informații legate de estimarea biomasei din oceane, măsurarea valurilor, a vitezei apei, a curenților, măsurarea topografică a adâncului oceanelor etc.



1. Navomodel

Navomodelul este o barcă care seamănă perfect cu una de jucărie , cu o telecomandă adaptată pentru distanțe de până la 600 de metri care ajută la pescuitul plantat (lasă momeala la o distanță foarte mare de mal astfel sporind șansele de a prinde un pește foarte mare) . Odată plantată momeala barca se întoarce goală înapoi la pescarul ce o controlează cu ajutorul telecomenzii. Barca este prevazută cu una sau mai multe cuve în care se pune momeală, și care la apăsarea unui buton aflat pe telecomandă se răstoarnă și eliberează montura alături de cârlig. Această bărcuță merge cu baterii de 12V care se pot încărca la priză.



##### c. Sisteme de avertizare(senzori și swingere);

În principal rolul unui semnalizator este de a semnala mușcătura peștelui printr-un semnal sonor mai mult sau mai puțin strident. Principiul de funcționare a majorității semnalizatoarelor este același doar detaliile de construcție și de design diferă de la o firmă producătoare la alta sau de la model la model. Pe lângă rolul de a semnala mușcătura, semnalizatorul are rol și de suport pentru lansetă și se montează în general pe rodpod sau pe suporți individuali.Lanseta se așează în locașul din partea superioară a semnalizatorului astfel încât firul să intre în contact cu rotița situată în fundul locașului   
Principiul de funcționare este destul de simplu și anume în momentul mușcăturii, firul care este în contact permanent cu rotița, antrenează în mișcare această rotiță de plastic sau metal. În cazul unui sistem de citire optic (printr-un optocuplor) rotița este prevăzută cu o tijă cu aripioare ce acționează o celulă fotoelectrică și declanșează un semnal acustic și un semnal optic prin intermediul a unui sau a două leduri luminoase. Mișcarea rotiței mai poate fi sesizată și printr-un traductor magnetic al sistemului de citire și anume printr-un senzor cu efect Hall.

Folosirea swingerelor de orice tip oferă o serie de avantaje importante :

* Se observă prompt mușcătura peștelui indiferent de condițiile de vânt în care se pescuiește, având posibilitatea de a varia greutatea swingerului.
* Majoritatea modelelor dispun de un sistem de atașare a indicatorilor luminoși, a starleților
* Un alt avantaj foarte mare este eliberarea din clips a firului în momentul înțepării, excluzând astfel riscul de a se produce o răsucire a bambinei în jurul lansetei ca și în cazul bambinei clasice care rămâne atașată la fir în timpul drilului.



***Avertizoarele electronice*** reprezintă vârful evoluției avertizoarelor folosite în pescuit. Datorită sensibilității lor, vor sesiza, vizual și sonor, chiar și cea mai mica mișcare a firului principal. Aceste dispozitive, indiferent de producător, par a fi identice, diferențierea făcându-se doar prin numărul de posibilități de avertizare. Dacă optăm pentru folosirea avertizoarelor electronice mai avem nevoie și de un dispozitiv pentru întinderea și mentinerea întinderii uniforme a firului. Pentru aceasta, putem folosi clasicele bambine sau swinghere special create. Avertizoarele electronice pot fi prezentate individual sau în seturi de două sau mai multe unități. La rândul lor, seturile pot fi compuse din mai multe avertizoare simple sau pot conține o centrală la care sunt conectate prin cablu sau unde radio. În cazul celor dotate cu o centrală ce funcționează prin unde radio, avertizoarele sunt conectate la un emițător care trimite un semnal către unitatea centrală indicând care dintre avertizoare a emis semnalul. Raza de recepție, în acest caz, este de aproximativ 50 de metri intre emitator si centrala. O dotare suplimentara a aparut la avertizoarele de ultima generatie, acestea semnaland ridicarea sau coborarea firului prin led-uri de culori diferite.

***I.3. Stabilirea unor criterii de inovare a dispozitivelor folosite în pescuit .***

Deoarece tehnologia avansează pe zi ce trece mai mult și mai mult , și în ramura pescuitului este nevoie/cerere de inovație. De aceea, m-am gândit să expun o serie de idei precum :

1. Un dispozitiv pentru măsurarea firului lansat.
2. Un dispozitiv care să stabilească greutatea peștelui.
3. Un dispozitiv care să estimeze poziția peștelui în apă (mai exact adâncimea și distanța de malul apei ).

Pe baza acestor idei se dorește ușurarea pescuitului staționar. Primul dispozitiv mă ajută ca de fiecare dată să arunc la aceeași distanța față de mal, lucru foarte important deoarece odată găsită(Darboux) , această distanță ideală care să aducă cât mai repede peștele la cârlig trebuie păstrată pentru a nu se pierde randamentul.

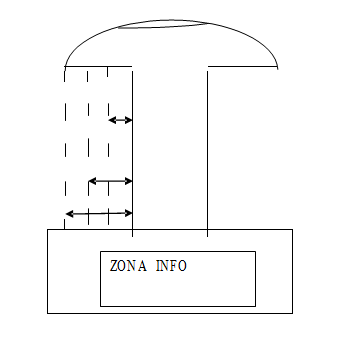
Al doilea dispozitiv mă ajută să știu cu ce am de a face, chiar dacă nu voi putea stabili exact cât cântărește peștele(deoarece sub apă, nu toți peștii au aceeași forță), voi clasifica peștele ca aparținând unui interval de greutate . În cazul în care peștele depășește o anumită greutate pescarul este dator să își schimbe atitudinea, dacă nu dorește să rateze acest caz excepțional.

Al treilea dispozitiv mă ajută să aflu distanța peștelui până la suprafața apei, de unde aș putea să calculez estimativ, în cât timp pot vedea peștele la suprafața apei (în momentul în care peștele este la suprafața apei , pescarului îi este mai ușor să îl scoată din apă).

**II. Modelarea matematică a inovațiilor.**

***II.1. Model matematic pentru stabilirea lungimii firului lansat;***

Pentru a calcula lungimea firului lansat, vom ține cont de specificațiile mulinetei, specificații care sunt furnizate de către producător, care se referă la cat fir de grosimea(mm) incape pe acea mulineta. Aceste date se află, la fiecare mulinetă, în ZONA INFO din schema de mai jos.

**

* Vom nota cu L dimensiunea firului care încape maxim pe tambur(metri).
* Vom nota cu g grosimea firului
* d din schema 1.1 este distanța dintre axul mulinetei și zona de ieșire din stânga sau din dreapta(raza tamburului).
* d1 din schema 1.1 este partea acoperită de firul pus de utilizatorul acesteia (firul de aceeași grosime pe toată mulineta).
* d2 din schema 1.1 este partea care rămâne acoperită de fir după aruncare.

Așadar, algoritmul care îmi calculează distanța de aruncare va fi :

-inițial calculez dimensiunea D=d2-d1;

-eu știu că pentru d încap L m de fir de grosime g așa că voi calcula câți m(metri) de fir de grosimea aleasă de utilizator.

-dacă știu câți metri de fir încap pe d , atunci pot calcula câți metri de fir sunt necesari pentru D. Așadar în funcție de rezultatul obținut anterior pot afla distanța de aruncare.

Pot calcula și viteza cu care se lansează:

-t0=momentul lansării

-t1=momentul impactului

d1-d0=

Pot calcula și forța de lansare:

-m= masa aruncata;

F=m\*dv/dt;

Pot calcula și accelerația:

a = dv/dt;

Pentru a calcula acestea , am nevoie de următoarele date pe care le-am aflat prin executarea unor serii de masuratori pe teren ce au constat intr-un numar de 20 de aruncari cu forte diferite.

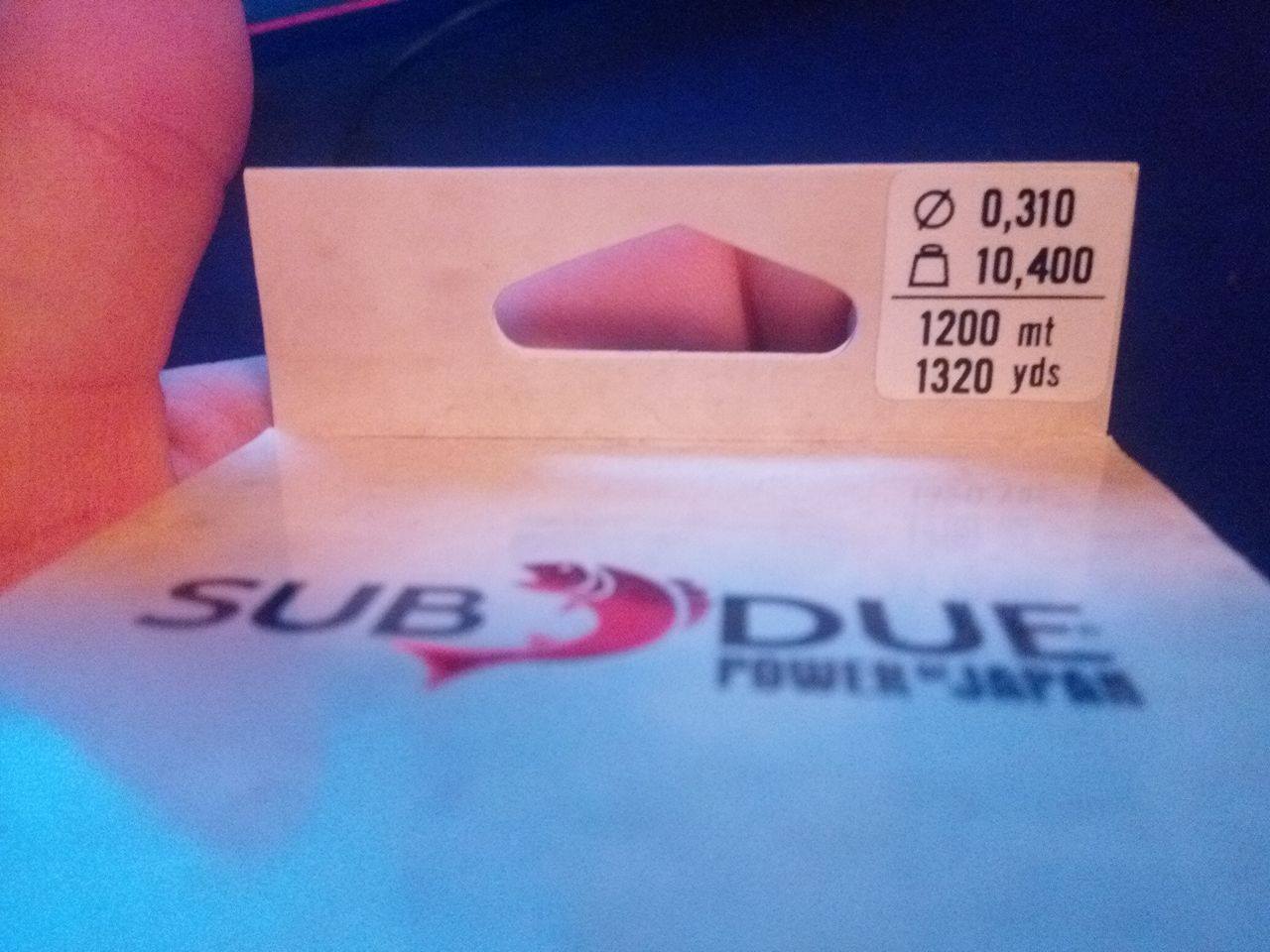
|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Nr | d/mm | d1/mm | d2/mm | L | g/mm | t1/s | m/g | Lungimea observată/m |
| 1 | 8 | 7.5 | 0.8 | 488.079 | 0.31 | 4.33 | 65.4 | 86m |
| 2 | 8 | 7.5 | 1.21 | 488.079 | 0.31 | 4.61 | 65.4 | 98m |
| 3 | 8 | 7.5 | 1.24 | 488.079 | 0.31 | 4.64 | 65.4 | 104m |
| 4 | 8 | 7.5 | 1.3 | 488.079 | 0.31 | 4.67 | 65.4 | 119m |
| 5 | 8 | 7.5 | 0.75 | 488.079 | 0.31 | 4.28 | 65.4 | 78m |
| 6 | 8 | 7.5 | 1.37 | 488.079 | 0.31 | 4.86 | 65.4 | 140m |
| 7 | 8 | 7.5 | 1.19 | 488.079 | 0.31 | 4.58 | 65.4 | 93m |
| 8 | 8 | 7.5 | 1.275 | 488.079 | 0.31 | 4.69 | 65.4 | 110m |
| 9 | 8 | 7.5 | 1.32 | 488.079 | 0.31 | 4.74 | 65.4 | 121m |
| 10 | 8 | 7.5 | 1.28 | 488.079 | 0.31 | 4.61 | 65.4 | 111m |
| 11 | 8 | 7.5 | 0.83 | 488.079 | 0.31 | 4.36 | 65.4 | 89m |
| 12 | 8 | 7.5 | 1.195 | 488.079 | 0.31 | 4.58 | 65.4 | 94m |
| 13 | 8 | 7.5 | 1.22 | 488.079 | 0.31 | 4.62 | 65.4 | 100m |
| 14 | 8 | 7.5 | 1.27 | 488.079 | 0.31 | 4.67 | 65.4 | 109m |
| 15 | 8 | 7.5 | 1.33 | 488.079 | 0.31 | 4.72 | 65.4 | 125m |
| 16 | 8 | 7.5 | 0.325 | 488.079 | 0.31 | 4.10 | 65.4 | 40m |
| 17 | 8 | 7.5 | 0.32 | 488.079 | 0.31 | 4.03 | 65.4 | 39m |
| 18 | 8 | 7.5 | 0.35 | 488.079 | 0.31 | 4.14 | 65.4 | 50m |
| 19 | 8 | 7.5 | 0.315 | 488.079 | 0.31 | 4.00 | 65.4 | 38m |
| 20 | 8 | 7.5 | 0.365 | 488.079 | 0.31 | 4.12 | 65.4 | 45m |

Pentru a obține aceste date am folosit mulineta:



Aceasta are următoarele specificații : încap 530m fir de 0.30mm,390m de 0.35mm , 265m fir de 0.40mm; dimensiunea razei este de 0.6cm .

Ca fir am folosit unul cu o grosime de 0.31mm.



**Pasul 1:** Știu rapoartele m/mm specifice mulinetei și voi calcula cât fir cu diametrul de 0.31 încape pe acest tambur.

Deoarece eu folosesc un fix cu o grosime de 0,31mm, grosime care nu este specificată pe tambur din fabrică, eu trebuie să calculez lungimea L în funcție de acest fir.

Pentru a face asta, inițial m-am interesat din diferite surse (forumuri și site-uri oficiale de producție a mulinetelor) despre cum se calculează aceste valori . Bineînțeles, nu am avut noroc să găsesc ceva concret, doar multe presupuneri, așa că m-am gândit sa fac eu asta.

Prima intrebare a fost cum să fac acest lucru?

Și am gândit astfel:

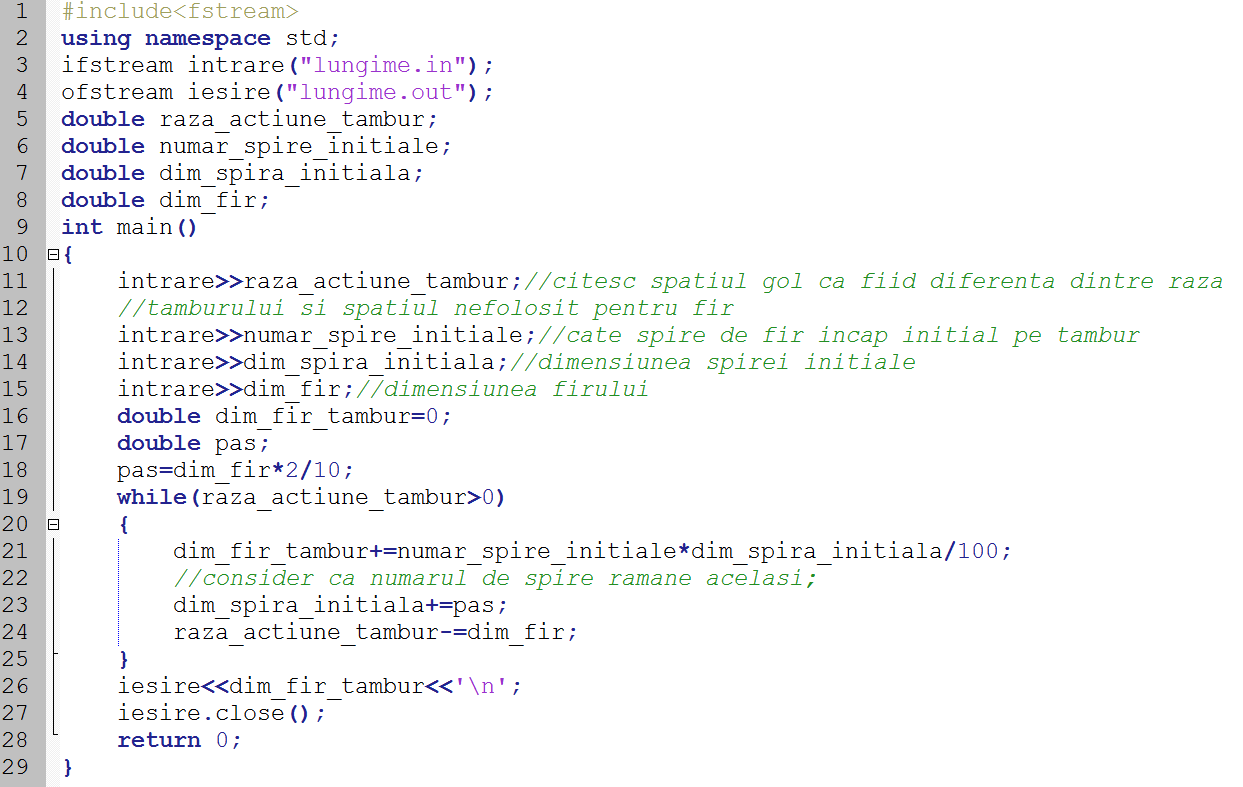
Ce știu?

* Știu diametrul mulinetei:6,7cm
* Știu care este raza de acțiune a tamburului (unde intră fir):0,8cm
* Știu grosimea firului
* Știu câte spire de fir intră pe tambur la o iterație(am calculat manual cu mare grijă câte spire complete intră în configurarea inițială a tamburului):119 spire
* Știu că o spiră are inițial 15cm

M-am gândit să concep un program care să îmi calculeze cât fir cu lungimea aceasta îmi intră pe tambur.

Programul are ca date de intrare, calculate de mine în prealabil, raza de acțiune a tamburului, numărul de spire inițiale, dimensiunea spirei inițiale și grosimea firului și va afișa cât fir cu această grosime încape pe tamburul meu.

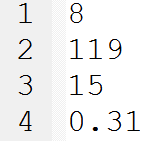
Am scris programul în C++. Aici este și codul:



Precizări cod:

În primele 2 linii am precizat că lucrez cu fișiere, apoi am declarat fișierele de intrare , respectiv ieșire. Pe liniile 5,6,7,8 am declarat datele de intrare. Pe liniile 11,13,14,15 am citit datele de intrare. Pe linia 16 am declarat și am inițializat cu 0 dimensiunea firului care încape pe tambur. Pe linia 17si 18 mi-am inițializat și am calculat pasul cu care se mărește dimensiunea spirei la fiecare pas(constanta). Apoi am spus că, atât timp cât tamburul nu este plin (adică nu am folosit 100% raza de acțiune a sa) pot să mai bag un rând de spire pe el. Acest procedeu constă în adăugarea la dimensiunea căutată a unei iterații complete. O iterație completă se produce când se introduc cele n spire cu dimensiunea spirei curentă. La fiecare iterație spira își mărește lungimea cu de doua ori grosimea firului. La sfârșitul iterației, din raza de acțiune am mai folosit ocupat o bucată egală cu grosimea firului și o scad din ea , după care pe linia 26 afișez lungimea cautată de mine .

Am rulat cu datele de intrare



Pe linia 1 am raza de acțiune a tamburului , în cazul meu 8 mm.

Pe linia 2 am numărul de spire inițiale (care este același la fiecare pas):119

Pe linia 3 am dimensiunea inițială a unei spire:15 cm

Pe linia 4 am grosimea firului

Am primit ca rezultat



Pe aceasta linie am numărul de metri de fir cu grosimea de 0,31mm care încape pe tamburul mulinetei mele.

**Pasul 2:** calculez distanța d pe care am folosit-o și la pasul 1 si am observat prin măsurare că este 0,8 cm.

**Pasul 3:** pot pune firul pe mulinetă și apoi pot executa aruncările necesare culegerii datelor externe.

Dupa ce am reușit să strâng datele de pe teren, mi-am dat seama că pentru a afla cât am aruncat, am nevoie de fapt de:

d1-starea inițială a tamburului

d2-starea finală a a tamburului

Având programul scris anterior este ușor să calculez distanța aruncării. De ce? Pentru că, inițial calculez cât încape pe tambur dacă este plin pentru d1 , apoi calculez cât fir încape pe tambur pentru d1-d2, rezultatul căutat fiind de fapt diferența dintre acestea.

Finalitatea acestui studiu constă într-un program prin care, dacă un pescar dorește să vadă exact cât a aruncat, în funcție de anumite date observabile, acesta să poată afla. În cazul în care acesta nu are idee cât este lungimea unei spire inițiale, când tamburul este gol, pentru a vedea distanța de aruncare este suficient să măsoare lungimea unei spire după aruncare, cu un metru(sau orice alt dispozitiv care poate indica aceasta distanță).

Am rescris codul pentru calcularea lungimii firului care încape pe tambur în MATLAB.

Apoi, am realizat programul pentru calcularea distanței aruncate.Trebuie specificat că, pentru calculul numărului de spire pot împarți înălțimea tamburului la grosimea firului. Acest număr rămâne constant pe parcursul calculării distanței căutate.

Formula pentru calculul numarului de spire:

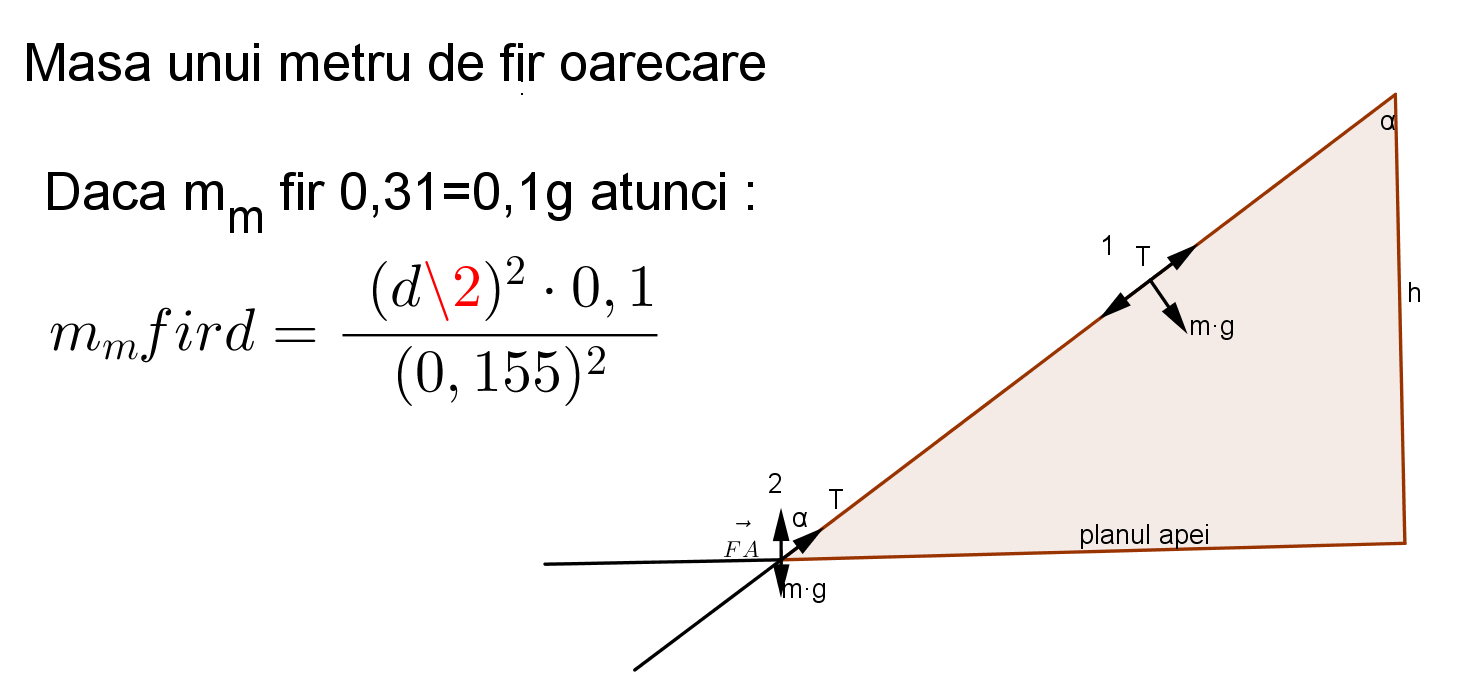
Nr\_spire=h\_tambur/grosime\_fir;

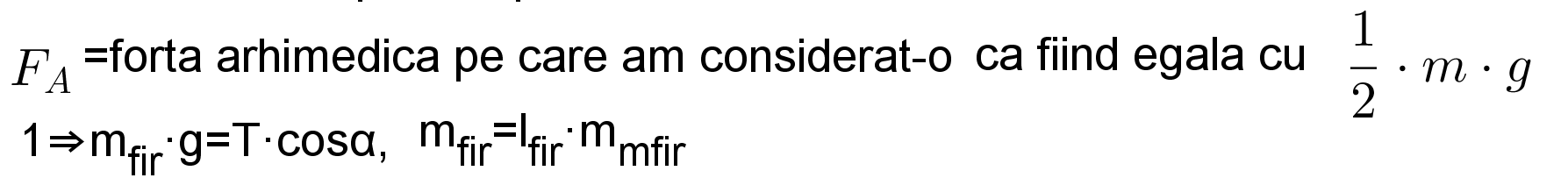
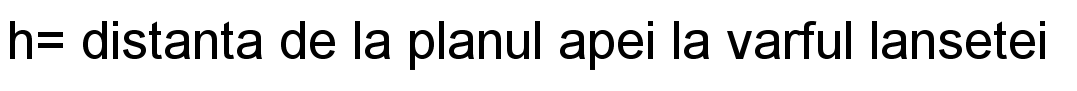
Dimensiunea spirei inițiale trebuie masurata dupa lansare utilizandu-se o rigla sau un metru. Aceasta se masoara fixandu-se firul intr-un punct in care acesta este tensinat apoi executandu-se o rotire a acestuia in jurul tamburului pana in acelasi punct. Distanta de la acest punct fixat pana la el reprezinta dimensiunea unei spire initiale.

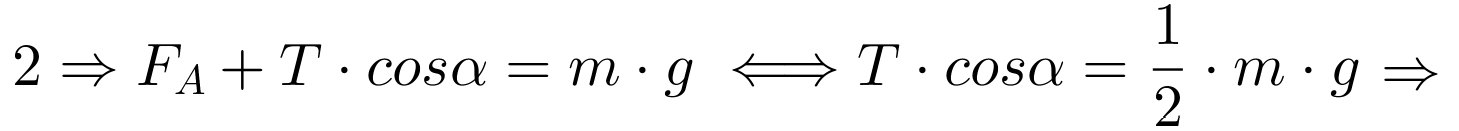
În concluzie, cu un minim de cunoștințe, se poate afla cât am aruncat exact cu lanseta.

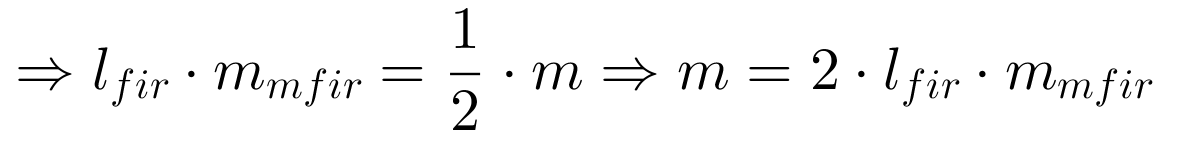
În continuare am realizat programul care să calculeze viteza de lansare a greutății,accelerația și forța de lansare.

**II.2. Model matematic pentru aprecierea greutății peștelui din “drill”;**











Pentru a putea să apreciem care este greutatea unui pește în drill ar trebui să ținem cont de un număr prea mare de factori. Dupa multe încercări de a ajunge la un rezultat apropiat de realitate, am reușit să realizez un model ideal pentru aflarea unei greutăți medii la care firul să stea întins.

Totuși, din studiul realizat pe parcursul derulării procesului de creare a tezei de licență am constatat că în funcție de forța unui pește din apă nu se poate măsura greutatea lui , asadar, pescarul de pe mal poate fi păcălit foarte ușor de un pește cu o putere mare și o greutate nu chiar atât de mare (acest lucru este firesc și întâlnit și la oameni care au forțe diferite). În lupta pentru supraviețuire a unui pește, el face orice pentru a putea scăpa , de aceea de cele mai multe ori intră în plantele de sub apă , pe care pescarul nu le poate observa, deci nici noi nu putem ști dacă pe fir sunt și plante la un moment dat sau nu, acestea schimbând semnificativ greutatea unui pește, aici referindu-mă la greutatea pe care o pot observa.

Pentru a calcula care este greutatea minimă necesară pentru ca firul să fie întins, adică în condiții optime pentru pescuitul la mare distanță vom avea nevoie de:

în primul rând vom avea nevoie de înălțimea dintre vârful lansetei și suprafața apei, distanță care poate fi măsurată sau intuită , doar de pescar deoarece, bineînțeles înălțimea malului apei poate să difere, înălțimea oamenilor, cu siguranță diferă și totodată diferă și lungimea lansetei. Odată măsurată aceasta înălțime în condiții optime, va putea fi utilizată în continuare pentru aflarea greutății minime necesare pentru a întinde firul;

în al doilea rând vom avea nevoie de greutatea unui metru liniar de fir utilizat de pescar;

Pentru a afla aceasta greutate am cântărit o rolă de 1200 m fir (0,31 mm diametru) sigilată și una goală:

-cea goală a cântărit 0,040kg = 40 g

-cea plină a cântărit 0,160kg = 160g

Deci 1200m fir de 0,31mm diametru cântărește 120g, de unde rezultă că un metru liniar de fir cu diametrul de 0,31mm cântărește exact 0,1 g.

Pentru a sări în ajutorul utilizatorului aplicației, ofer o formulă care calculează exact care este masa unui metru liniar de fir în funcție de diametrul lui:

msD=((D/2)^2 \* 0.1)/(0.155)^2

Am scris codul în Python:

i=0.01  
**while** i<=1:  
 dimi=((i/2)\*\*2\*0.1)/((0.155)\*\*2)  
 print(**"Firul cu diametrul de "**,i,**" are greutatea "**,dimi)  
 i=i+0.01

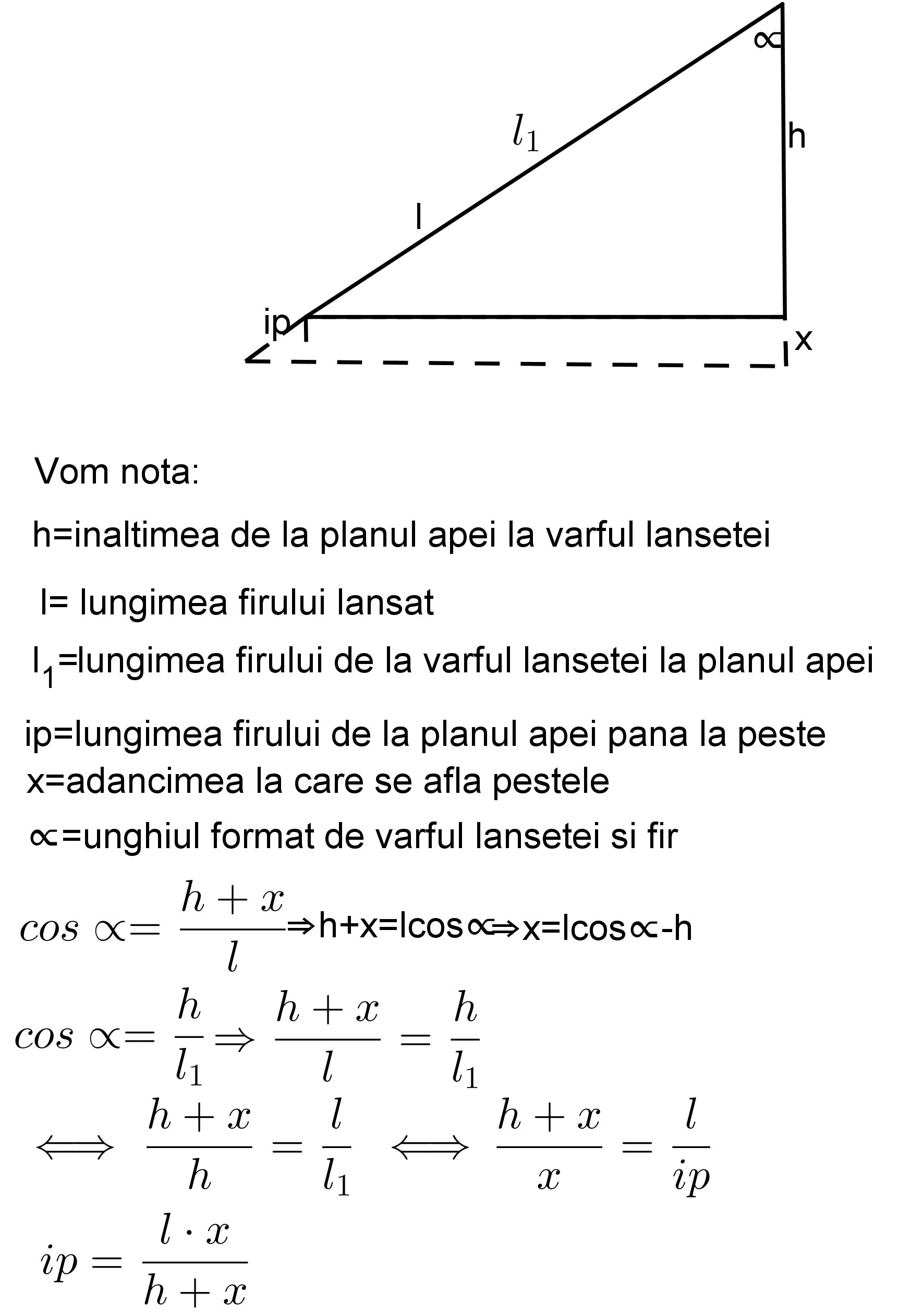
Și am primit ca rezultat pentru orice fir existent greutatea unui metru liniar de fir cu diametrul respectiv oferit de utilizator.

Odată cu aflarea acestor date, vom putea afla greutatea minima a monturii pentru ca firul să fie tensionat în condiții normale.

Formula de calcul este următoarea:

masaFirIntins=2\*lungimeafirului\*masamedieafirului(un metru de fir)

Astfel putem observa cum distanța de aruncare a monturii afectează modul în care se întinde firul.

**II.3. Model matematic pentru aproximarea poziției în care se află peștele de la capătul** **firului;**

Important, în ceea ce privește poziția peștelui în apă, este adâncimea la care el se află la un moment dat. Această adâncime este atât de importantă deoarece fiecare pescar în parte știe, cu siguranță, că un pește care “a luat aer” este ca și prins, deoarece în momentul în care el a scos capul din apă și a luat o gură sănătoasă de aer, puterile sale scad semnificativ.

Pentru a sprijini această idee cum că peșștele scos la suprafață este o pradă mai ușoar, m-am gândit că ar fi folositor fiecarui pescar să știe cât a fost de aproape de prinderea unei capturi eventual scăpate, sau cât durează în medie până când un pește ajunge la suprafață în funcție de greutatea lui.

Așadar, m-am gândit să calculez cât de aproape de suprafață a ajuns peștele în timpul drilului (cât fir mai trebuia să mulinez până când peștele să ajungă la aer), pe baza unor date observate anterior. Pentru a calcula acest lucru este suficient să calculez care este lungimea ip din modelul realizat anterior.

Pentru a calcula cât dureaza în medie ca un pește să ajungă să ia contact cu aerul, în funcție de forța fiecărui pescar și tehnica utilizată de el, acesta trebuie să utilizeze aplicația de un număr suficient de ori pentru diverși pești pentru ca la un moment dat să poata aproxima durata medie de obosire a unui pește. În momentul în care am un număr suficient de mare de teste făcute, pot spune că aproximarea este exactă.

**III. Implementarea modelelor descrise în capitolul II.**

**III.1. Cod MATLAB de calcul numeric pentru modelul de la II.1.**

Pentru modelul descris în cadrul capitolului II subcapitolul 1, am realizat codul in Matlab pentru:

1.Cât fir încape pe o mulinetă? (orice mulinetă , orice specificații)

Pentru a calcula aceasta distanță am avut nevoie să cunosc înainte raza de acțiune a tamburului, numărul de spire inițiale, dimensiunea spirei inițiale și dimensiunea firului utilizat, după cum am explicat anterior in capitolul sus numit.

Codul Matlab explicat:

clear all;

#am șters datele reținute în buffer înainte de începerea rulării programului meu.

raza\_actiune\_tambur=input('Care este raza de actiune a tamburului?');

numar\_spire\_initiale=input('Care este numarul initial de spire?');

dim\_spira\_initiala=input('Care este dimensiunea unei spire initiale?');

dim\_fir=input('Care este dimensiunea firului utilizat?');

#am citit de la tastatura cele 4 date de intrare.

lungimea\_fir\_tambur=0;

pas = dim\_fir \* 2 / 10;

#la fiecare pas lungimea spirei de mareste cu de 2 ori grosimea firului in centimetre de aici formula de mai sus pentru calculul unui pas (firesc diametrul are acelasi comportament, adica se mareste la fiecare iteraie completa cu de doua ori grosimea firului)

dim\_spira=dim\_spira\_initiala;

while raza\_actiune\_tambur>0

lungimea\_fir\_tambur=lungimea\_fir\_tambur+ numar\_spire\_initiale\*dim\_spira\_initiala / 100;

dim\_spira\_initiala =dim\_spira\_initiala+ pas;

raza\_actiune\_tambur =raza\_actiune\_tambur- dim\_fir;

end

disp(lungimea\_fir\_tambur);

#calculez lungimea firului care incape pe mulineta pentru datele de intrare citite si afisez pe ecran.

dim\_umpluta=input('Care este raza ocupata cu fir dupa aplicarea firului pe tambur?');

#citesc de la tastatura care este raza ocupata cu fir dupa ce am aplicat firul pe tambur pentru a sti exact cat am pus. Apoi utilizand acelasi algoritm ca mai sus calculez cat fir am pe mulineta.

dim\_spira\_initiala=dim\_spira;

lungimea\_fir\_tambur\_plin=0;

while dim\_umpluta>0

lungimea\_fir\_tambur\_plin=lungimea\_fir\_tambur\_plin+ numar\_spire\_initiale\*dim\_spira\_initiala / 100;

dim\_spira\_initiala =dim\_spira\_initiala+ pas;

dim\_umpluta =dim\_umpluta- dim\_fir;

end

disp(lungimea\_fir\_tambur\_plin);

#afisez pe ecran cat fir are mulineta respectiva.

dim\_u=input('Care este raza ocupata cu fir dupa aruncare?');

#citesc care este raza ocupata cu fir dupa executarea aruncarii si calculez cu acelasi algoritm cat fir este acum pe mulineta.

dim\_spira\_initiala=dim\_spira;

lungimea\_fir\_tambur\_final=0;

while dim\_u>0

lungimea\_fir\_tambur\_final= lungimea\_fir\_tambur\_final+ numar\_spire\_initiale\*dim\_spira\_initiala / 100;

dim\_spira\_initiala =dim\_spira\_initiala+ pas;

dim\_u =dim\_u- dim\_fir;

end

disp(lungimea\_fir\_tambur\_final);

#afisez pe ecran cat fie este pe mulineta respectiva dupa aruncare

lungime\_fir\_aruncat=lungimea\_fir\_tambur\_plin-lungimea\_fir\_tambur\_final;

#calculez si afisez pe ecran diferenta dintre cat fir am avut inainte de aruncare si cat am dupa aruncare.Aceasta diferenta ne dezvaluie cat fir am aruncat defapt.

fprintf('Bravo, ai aruncat: %f\n',lungime\_fir\_aruncat);

timp1=input('Care este timpul pe care l-a parcurs montura pana la impactul cu apa?');

#citesc cat timp a durat executarea aruncarii

timp0=0;

#initial t0=0

dist0=0;

#initial distant aruncata este 0

dist1=lungime\_fir\_aruncat;

#distanta aruncata este egala cu lungimea firului lansat

viteza=(dist1-dist0)/(timp1-timp0);

#viteza este d/dt unde d este diferenta dintre dist1 si dist0 iar dt este diferenta dintre timp1 si timp0;

fprintf('Viteza pe care ai reusit sa o impui greutatii este: %f m/s\n',viteza);

#afisez viteza de aruncare

acceleratia=viteza/timp1;

fprintf('Acceleratia masurata in timpul aruncarii este: %f\n',acceleratia);

#calculez si afisez acceleratia masurata in timpul aruncarii.

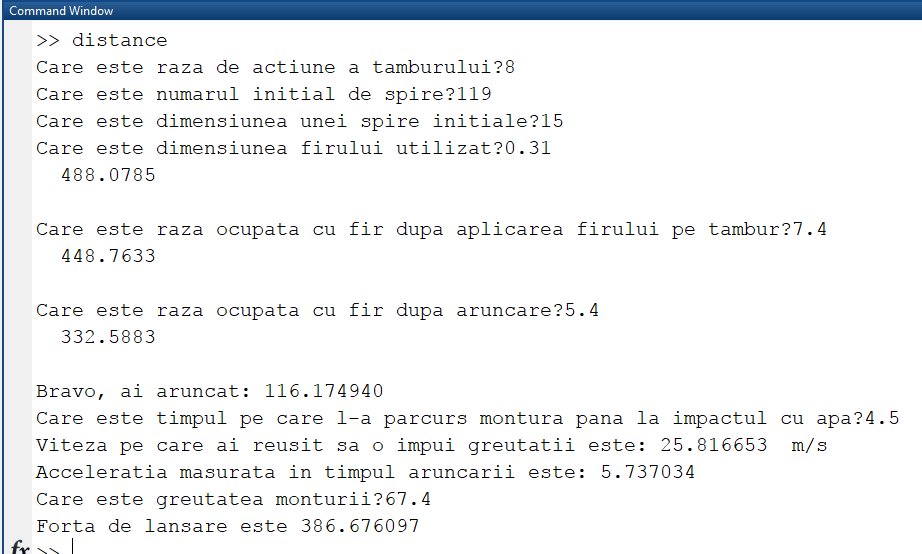
masa\_aruncata=input('Care este greutatea monturii?');

forta\_lansare=masa\_aruncata\*acceleratia;

fprintf('Forta de lansare este %f\n',forta\_lansare);

#am citit masa monturii pe care am aruncat-o pentru a putea calcula forta de lansare a monturii.

Programul ruleaza pentru un set de date in felul urmator:



**III.2. Tratarea grafică, în MATLAB , a modelului descris la II.2.**

Pentru modelul descris în capitolul II subcapitolul al doilea am tratat grafic:

1.Evoluția greutății unui metru de fir în funcție de grosimea acestuia

clear all;

#am golit bufferul pentru a executa programul in siguranta fara a fi afectat de date anterioare

ax1 = subplot(1,1,1);

#am definit planul in care se va desena graficul

x = logspace(0,1);

#am definit functia x ca fiind o functie ce creste logaritmic grosimile firului

y = (x/2.^2\*0.1)/((0.155).^2);

#am definit functia y ca fiind calculul greutatii unui metru de fir liniar cu grosimea x

plot(ax1,x,y,'-s')

#am desenat functiile pe grafic

grid on

title(ax1,'Graficul greutatii unui metru liniar de fir')

#am adaugat un titlu pentru graficul meu

ylabel(ax1,'masa metru fir')

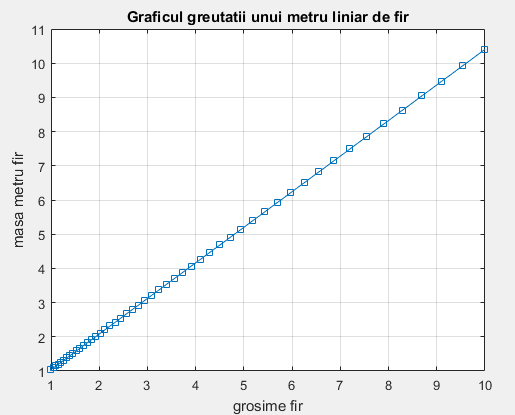
#axa Oy reprezinta evolutia masei unui metru de fir asadar i-am oferit un nume sugestiv

xlabel(ax1,'grosime fir')

#axa Ox reprezinta evolutia grosimii unui metru de fir asadar i-am oferit un nume sugestiv

In urma rularii acestui program se poate observa cum odata cu grosimea firului creste si greutatea acestuia liniar.

Graficul rezultat in urma rularii acestui program este:



2.Evoluția greutății minime ca firul să fie întins în funcție de masa firului și lungimea sa.

Prima dată am realizat codul pentru calculul greutății minime ca firul să stea întins după cum urmează:

clear all;

distanta\_aruncata=input('Care este distanta la care ati aruncat?\n');

#am citit care este distanta pe care am aruncato(distanta pe care o pot afla cu algoritmul prezentat la II 1).

grosime\_fir=input('Care este grosimea firului utilizat?\n');

#citesc grosimea firului

greutate\_metru\_fir=(((grosime\_fir/2)^2)\*0.1)/((0.155)^2);

#calculez greutatea unui metru de fir cu acea greutate

masa\_fir\_intins=2\*distanta\_aruncata\*greutate\_metru\_fir

#calculez masa minima necesara ca firul sa fie intins si o afisez

fprintf('Greutatea pentru care firul sta intins la distanta %f m este de %f g\n',distanta\_aruncata,masa\_fir\_intins);

Dupa realizarea codului pentru calculul greutății medii pentru ca firăul s fie întins am făcut un grafic în care se poate vedea cum crește această greutate calculată anterior raportat la lungimea aruncării, grosimea firului și masa acestuia.

clear all;

ax1 = subplot(1,1,1);

x = logspace(0,1);

#grosimea firului ia valori intre 0 si 1

distanta=unifrnd(50,150);

#am definit functia distanta ca fiind o reprezentare uniforma in intervalul 50 100

y = (x/2.^2\*0.1)/((0.155).^2);

#y este functia care calculeaza masa unui metru de fir

z = 2\*distanta\*y;

#z este functia care calculeaza greutatea minima pentru o distanta si un anumit fir

plot3(ax1,x,y,z,'-s')

#am asociat cele 3 functii x,y si z graficului cu numele ax1(un grafic 3D)

grid on;

title(ax1,'Graficul greutatii minime de aruncare')

#am oferit graficului un nume sugestiv

ylabel(ax1,'masa metru fir')

#axa Ox reprezinta evolutia masei unui metru de fir asadar i-am oferit un nume sugestiv

xlabel(ax1,'grosime fir')

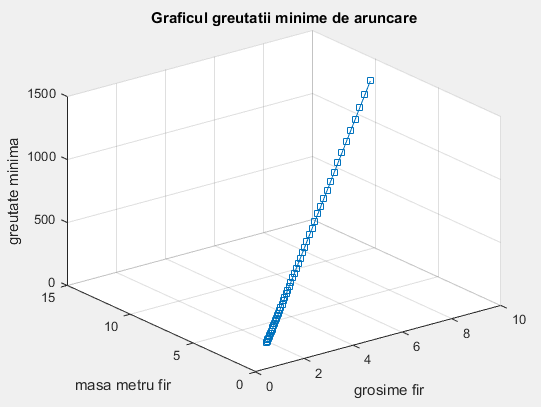
#axa Ox reprezinta evolutia grosimii unui metru de fir asadar i-am oferit un nume sugestiv

zlabel(ax1,'greutate minima')

#axa Oz reprezinta evolutia greutatii minime necesare ca firul sa fie intins, asadar i-am oferit un nume sugestiv.

În urma rulării acestui program se poate observa cum, odată cu lungimea distanței aruncate și cu greutatea unui metru de fir, crește și greutatea minimă pentru ca firul să fie întins în condiții de echilibru.

Graficul rezultat în urma rulării acestui program este:



**III.3. Utilizarea Simulink-ului pentru realizarea în MATLAB a modelului evidențiat la II.3.**

Pentru început, pentru modelul evidențiat la II.3, am construit codul în Matlab, cod ce poate fi folosit de un viitor utilizator al aplicației mele.

Acest cod este:

clear all;

#am șters istoricul aplicațiilor rulate înainte pentru a nu afecta rularea actuală

h=input('Care este înălțimea de la planul apei la vârful lansetei?');

#cer utilizatorului înălțimea pe care acesta trebuie să o observe la locul în care pescuiește

l=input('Care este lungimea firului din apă?');

#citesc ca dată de intrare și care este lungimea firului din apă la momentul în care pescarul interogheaza aplicația(aceasta lungime poate fi calculată dupa modelul prezentat la II.1);

alpha=input('Care este mărimea în grade a unghiului format de vârful lansetei și planul apei?');

#citesc și unghiul format de planul firului cu cel al lansetei

x=l\*cos(alpha)-h;

#calculez care este adâncimea maximă la care se poate afla peștele

fprintf('Pestele se afla la adancimea de %f\n',x);

ip=(l\*x)/(h+x);

#calculez care este distanța dintre pește și aer la momentul în care utilizatorul interoghează aplicația.

fprintf('Pentru a ajunge la aer peștele mai are nevoie de %f\n',ip);

Pentru a evidenția la un moment dat al drilului cu peștele, o imagine de ansamblu asupra a ceea ce se întâmplă de fapt, am utilizat Simulink.

SIMULINK este un mediu pentru modelarea, analiza şi simularea unui mare număr de sisteme fizice şi matematice.

Ca extensie opţională a pachetului de programe MATLAB, SIMULINK oferă o interfaţă grafică cu utilizatorul pentru realizarea modelelor sistemelor dinamice reprezentate în schema bloc. O bibliotecă vastă, cuprinzând cele mai diferite blocuri stă la dispoziţia utilizatorului. Aceasta permite modelarea rapidă şi clară a sistemelor, fără a fi necesară scrierea măcar a unui rând de cod de simulare.

Modelele realizate sunt de natură grafică, iar pe lângă numeroase alte avantaje SIMULINK oferă şi posibilitatea de documentare şi de tipărire a rezultatelor la imprimantă. Rezultatele simulării unui sistem pot fi urmărite chiar în timp ce se desfăşoară simularea, pe un osciloscop reprezentat într-o fereastră a ecranului

Simulink este o colecție de funcții MATLAB, organizate într-un așa numit toolbox al sistemului de programare menționat. Simulink aduce în plus funcționalități specifice analizei și sintezei sistemelor dinamice, păstrând în același timp toate caracteristicile și funcționalitățile sistemului MATLAB.

Exista două faze logice de utilizare a toolboxului. Într-o primă fază, se definește sau se apelează un model de sistem existent. Analiza acestui model face obiectul fazei a doua.

Toolboxul SIMULINK folosește o clasă de ferestre denumite “diagrame”. În astfel de ferestre este creat modelul sistemului.

Așadar, aș dori să realizez un model care să ajute la imaginarea relației dintre pește si pescar în diferite momente, când acesta dorește să vadă evoluția unui anumit drill. De aceea, am încercat să identific care sunt elementele care fac parte din acest model, și, am decis că acestea ar fi: peștele , firul, apa , lanseta și pescarul.

Peștele l-am construit ca fiind pasul de început al modelului, cel de la care pleacă(începe) evoluția acestuia.

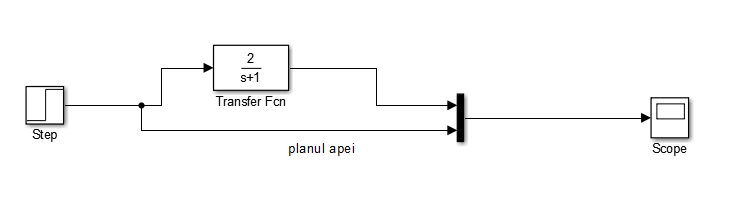
Firul, fiind cel care ajută la schimbarea mediului din care face parte peșștele, l-am modelat ca fiind o funcție de transfer dintre două medii.

Planul apei este modelat prin relația dintre pește , fir și lansetă.

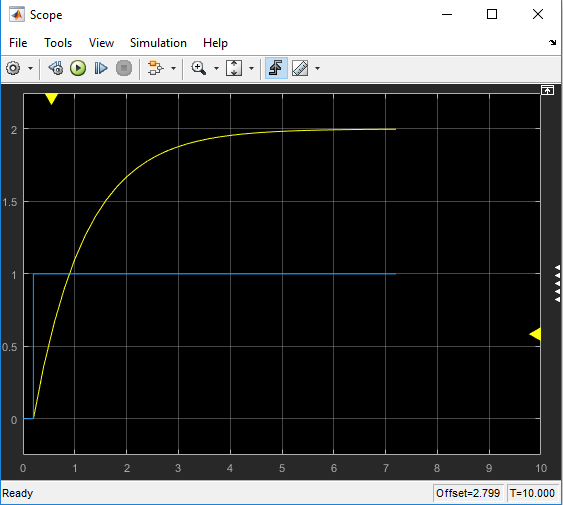
Lanseta este modelată ca un multiplexor care primește ca semnal funcția de transfer si relația dintre planul apei și pește oferă pescarului un rezultat.

Pescarul l-am definit ca fiind scopul modelului, cel care beneficiază de un grafic în care se observă situația din momentul în care acesta interoghează aplicația.

Modelul aplicației este:



Rezultatul calculat în urma rulării aplicației generată de model este:



În acest grafic se poate observa planul apei cu albastru și firul cu galben , în punctul din care pleacă cele două, se consideră că se află peștele. În punctul în care se termină planul apei, se consideră că este malul, unde este și pescarul. În punctul în care se termină firul, se consideră că se află vârful lansetei. Distanța dintre aceste două puncte, este de fapt h, din modelul prezentat la II.3.

**IV.Implementarea unei aplicații care să ajute la utilizarea într-un mod mai ușor a celor prezentate la capitolul II**

**IV.1.Interfața aplicației.**

Pentru a face mai ușoară utilizarea aplicațiilor realizate de mine anterior, am ales să fac o aplicație de timpul „Form Aplication” în Csharp. Această aplicație este făcută special pentru desktop și, odata instalată poate fi utilizată de către cei interesați să afle mai multe detalii teoretice despre subiectele atinse de mine în această lucrare.

Aplicația dispune de o interfață grafică simplistă care oferă rezultate rapide pentru diverse întrebari, în funcție de datele observate de fiecare utilizator în parte.

În cadrul acesteia, am înglobat, legând între ele toate subiectele in cadrul cărora am ajuns la un rezultat. Așadar, aplicația este capabilă să răspundă la întrebări după cum urmeaza:

1.Cât fir încape pe mulineta mea?

2.Cât fir am pus pe mulineta mea?

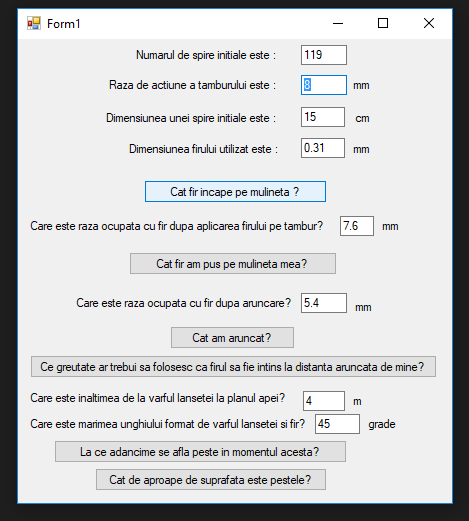
3.Cât am aruncat?

4.Ce greutate ar trebui să folosesc ca firul să fie întins la distanța precizată?

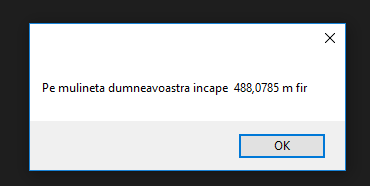
5.La ce adâncime se află peștele?

6.Cât de aproape de suprafață este peștele?

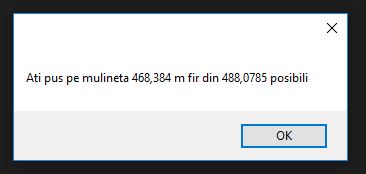
Interfața aplicației este următoarea:



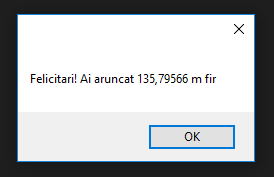
Pentru întrebarea „Cât fir încape pe mulinetă?” cu date de intrare numărul de spire inițiale, raza de acțiune a tamburului, dimensiunea unei spire inițiale și dimensiunea firului utilizat, voi primi un rezultat de genul:



Pentru întrebarea „Cât fir am pus pe mulineta mea?” voi avea nevoie ca și data de intrare și de raza ocupată cu fir după aplicarea acestuia pe tambur. Răspunsul primit va fi de genul:

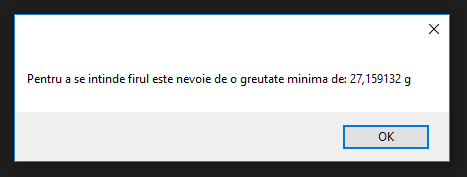


Pentru intrebarea „Cât am aruncat?” voi avea nevoie de raza ocupată cu fir dupa realizarea aruncării. Răspunsul primit va fi de forma:



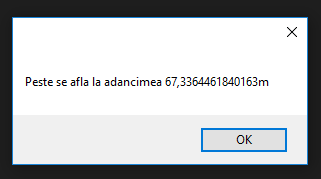
Pentru întrebarea „Ce greutate ar trebui să folosesc ca firul să fie întins la distanța aruncată de mine?” voi primi un răspuns care mă ajută să îmi dau seama, ca pescar , în cazul în care pescuiesc la plantat (adică duc cu barca montura la o anumită distanță și o las acolo) și doresc ca montura să stea pe loc și să se întindă firul, care ar fi greutatea minimă pe care ar trebui să o cântărească montura pentru a putea planta în locul în care am aruncat.

Răspunsul primit va fi de forma:



Pentru întrebarea „La ce adâncime se află peștele în momentul acesta?” în cadrul aplicației voi primi un răspuns valabil pentru cazul inițial la t0=0s, adica exact la începerea drilului deoarece datele introduse anterior descriu o aruncare, nu un anumit moment al drilului.

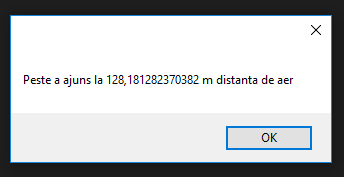
Răspunsul la aceasta întrebare poate să ajute pescarul să știe cât are de muncă pentru a scoate peștele de la adâncimea respectivă și va fi de forma:



Pentru întrebarea „Cât de aproape de suprafață se află peștele?” răspunsul este de asemenea în momentul t0=0s cand voi avea în apă foarte mult fir.

De aceea, valoarea este aproape de distanța aruncata,l1 din modelul prezentat la II.3 având o valoare foarte mică.

Răspunsul este de forma:



**IV.2.Elemente de programare și implementarea propriuzisă a aplicației.**

După cum am specificat și în subcapitolul anterior, pentru realizarea aplicației am utilizat ca model un form application. Programul utilizat pentru a scrie codul a fost Visual Studio Comunity 2015.

În cadrul unei aplicații de tipul Form Application există mai multe elemente printre care: clasa de bază a aplicației și clasa Formului care se împarte în design și codul propriuzis din spatele designu-lui. Primul pas în cadrul realizării unei astfel de aplicații este construcția designu-lui aplicației prin utilizarea unor elemente grafice specifice din intrumentele oferite de mediul Visual Studo. Acest design, odată realizat, generează codul pentru funcționalitatea aplicației, cod care nu face practic nimic în faza incipientă, fiind necesar ca programatorul aplicației să îl gândească pe parcurs pentru a oferi viață funcțiilor generate. Pentru a se putea rula aplicația grafică din clasa de bază a aplicației se apeleaza codul din spatele designu-lui.

Eu am urmărit atent, pas cu pas, ca elementele utilizate de mine să fie legate între ele, evitând astfel eventualele repetiții inutile care să creeze neplăceri utilizatorilor aplicației.

Programul de bază are forma:

using System;

using System.Windows.Forms;

namespace FishingAplication

{

static class Program

{

[STAThread]

static void Main()

{

Application.EnableVisualStyles();

Application.SetCompatibleTextRenderingDefault(false);

Application.Run(new Form1());

}

}

}

În cadrul acestuia se poate observa că se apelează o serie de funcții, ale bibliotecii System. Windows. Forms, care să ofere posibilitatea de vizualizare grafică a aplicației.

Prin apelul Application.Run(new Form1()), aplicația rulează designul și codul din spatele acestuia pentru form-ul cu numele Form1, creând o noua sesiune pentru această rulare.

Design-ul aplicației este de forma:

namespace FishingAplication

{

partial class Form1

{

private System.ComponentModel.IContainer components = null;

protected override void Dispose(bool disposing)

{

if (disposing && (components != null))

{

components.Dispose();

}

base.Dispose(disposing);

}

private void InitializeComponent()

{

this.label1 = new System.Windows.Forms.Label();

this.label2 = new System.Windows.Forms.Label();

this.label3 = new System.Windows.Forms.Label();

this.label4 = new System.Windows.Forms.Label();

this.textBox1 = new System.Windows.Forms.TextBox();

this.textBox2 = new System.Windows.Forms.TextBox();

this.textBox3 = new System.Windows.Forms.TextBox();

this.textBox4 = new System.Windows.Forms.TextBox();

this.label5 = new System.Windows.Forms.Label();

this.label6 = new System.Windows.Forms.Label();

this.label7 = new System.Windows.Forms.Label();

this.button1 = new System.Windows.Forms.Button();

this.label8 = new System.Windows.Forms.Label();

this.label9 = new System.Windows.Forms.Label();

this.textBox5 = new System.Windows.Forms.TextBox();

this.label10 = new System.Windows.Forms.Label();

this.button2 = new System.Windows.Forms.Button();

this.label11 = new System.Windows.Forms.Label();

this.textBox6 = new System.Windows.Forms.TextBox();

this.label12 = new System.Windows.Forms.Label();

this.button3 = new System.Windows.Forms.Button();

this.button4 = new System.Windows.Forms.Button();

this.textBox7 = new System.Windows.Forms.TextBox();

this.textBox8 = new System.Windows.Forms.TextBox();

this.label13 = new System.Windows.Forms.Label();

this.label14 = new System.Windows.Forms.Label();

this.label15 = new System.Windows.Forms.Label();

this.label16 = new System.Windows.Forms.Label();

this.button5 = new System.Windows.Forms.Button();

this.button6 = new System.Windows.Forms.Button();

this.SuspendLayout();

//

// label1

//

this.label1.AutoSize = true;

this.label1.Location = new System.Drawing.Point(88, 39);

this.label1.Name = "label1";

this.label1.Size = new System.Drawing.Size(177, 13);

this.label1.TabIndex = 0;

this.label1.Text = "Raza de actiune a tamburului este : ";

//

// label2

//

this.label2.AutoSize = true;

this.label2.Location = new System.Drawing.Point(333, 39);

this.label2.Name = "label2";

this.label2.Size = new System.Drawing.Size(23, 13);

this.label2.TabIndex = 1;

this.label2.Text = "mm";

//

// label3

//

this.label3.AutoSize = true;

this.label3.Location = new System.Drawing.Point(115, 9);

this.label3.Name = "label3";

this.label3.Size = new System.Drawing.Size(150, 13);

this.label3.TabIndex = 2;

this.label3.Text = "Numarul de spire initiale este : ";

//

// label4

//

this.label4.AutoSize = true;

this.label4.Location = new System.Drawing.Point(85, 72);

this.label4.Name = "label4";

this.label4.Size = new System.Drawing.Size(180, 13);

this.label4.TabIndex = 3;

this.label4.Text = "Dimensiunea unei spire initiale este : ";

//

// textBox1

//

this.textBox1.Location = new System.Drawing.Point(283, 36);

this.textBox1.Name = "textBox1";

this.textBox1.Size = new System.Drawing.Size(46, 20);

this.textBox1.TabIndex = 4;

this.textBox1.Text = "8";

this.textBox1.TextChanged += new System.EventHandler(this.textBox1\_TextChanged);

//

// textBox2

//

this.textBox2.Location = new System.Drawing.Point(283, 6);

this.textBox2.Name = "textBox2";

this.textBox2.Size = new System.Drawing.Size(46, 20);

this.textBox2.TabIndex = 5;

this.textBox2.Text = "119";

this.textBox2.TextChanged += new System.EventHandler(this.textBox2\_TextChanged);

//

// textBox3

//

this.textBox3.Location = new System.Drawing.Point(283, 68);

this.textBox3.Name = "textBox3";

this.textBox3.Size = new System.Drawing.Size(44, 20);

this.textBox3.TabIndex = 6;

this.textBox3.Text = "15";

this.textBox3.TextChanged += new System.EventHandler(this.textBox3\_TextChanged);

//

// textBox4

//

this.textBox4.Location = new System.Drawing.Point(283, 99);

this.textBox4.Name = "textBox4";

this.textBox4.Size = new System.Drawing.Size(44, 20);

this.textBox4.TabIndex = 7;

this.textBox4.Text = "0.31";

this.textBox4.TextChanged+=newSystem.EventHandler(this.textBox4\_TextChanged)

//

// label5

//

this.label5.AutoSize = true;

this.label5.Location = new System.Drawing.Point(335, 72);

this.label5.Name = "label5";

this.label5.Size = new System.Drawing.Size(21, 13);

this.label5.TabIndex = 8;

this.label5.Text = "cm";

//

// label6

//

this.label6.AutoSize = true;

this.label6.Location = new System.Drawing.Point(108, 103);

this.label6.Name = "label6";

this.label6.Size = new System.Drawing.Size(159, 13);

this.label6.TabIndex = 9;

this.label6.Text = "Dimensiunea firului utilizat este : ";

//

// label7

//

this.label7.AutoSize = true;

this.label7.Location = new System.Drawing.Point(333, 103);

this.label7.Name = "label7";

this.label7.Size = new System.Drawing.Size(23, 13);

this.label7.TabIndex = 10;

this.label7.Text = "mm";

//

// button1

//

this.button1.Location = new System.Drawing.Point(126, 141);

this.button1.Name = "button1";

this.button1.Size = new System.Drawing.Size(183, 23);

this.button1.TabIndex = 11;

this.button1.Text = "Cat fir incape pe mulineta ?";

this.button1.UseVisualStyleBackColor = true;

this.button1.Click += new System.EventHandler(this.button1\_Click);

//

// label8

//

this.label8.AutoSize = true;

this.label8.Location = new System.Drawing.Point(33, 167);

this.label8.Name = "label8";

this.label8.Size = new System.Drawing.Size(0, 13);

this.label8.TabIndex = 12;

//

// label9

//

this.label9.AutoSize = true;

this.label9.Location = new System.Drawing.Point(10, 180);

this.label9.Name = "label9";

this.label9.Size = new System.Drawing.Size(299, 13);

this.label9.TabIndex = 13;

this.label9.Text = "Care este raza ocupata cu fir dupa aplicarea firului pe tambur?";

//

// textBox5

//

this.textBox5.Location = new System.Drawing.Point(322, 177);

this.textBox5.Name = "textBox5";

this.textBox5.Size = new System.Drawing.Size(34, 20);

this.textBox5.TabIndex = 14;

this.textBox5.Text = "7.6";

this.textBox5.TextChanged += new System.EventHandler(this.textBox5\_TextChanged);

//

// label10

//

this.label10.AutoSize = true;

this.label10.Location = new System.Drawing.Point(362, 180);

this.label10.Name = "label10";

this.label10.Size = new System.Drawing.Size(23, 13);

this.label10.TabIndex = 15;

this.label10.Text = "mm";

//

// button2

//

this.button2.Location = new System.Drawing.Point(111, 213);

this.button2.Name = "button2";

this.button2.Size = new System.Drawing.Size(208, 23);

this.button2.TabIndex = 16;

this.button2.Text = "Cat fir am pus pe mulineta mea?";

this.button2.UseVisualStyleBackColor = true;

this.button2.Click += new System.EventHandler(this.button2\_Click);

//

// label11

//

this.label11.AutoSize = true;

this.label11.Location = new System.Drawing.Point(56, 257);

this.label11.Name = "label11";

this.label11.Size = new System.Drawing.Size(221, 13);

this.label11.TabIndex = 17;

this.label11.Text = "Care este raza ocupata cu fir dupa aruncare?";

//

// textBox6

//

this.textBox6.Location = new System.Drawing.Point(283, 254);

this.textBox6.Name = "textBox6";

this.textBox6.Size = new System.Drawing.Size(46, 20);

this.textBox6.TabIndex = 18;

this.textBox6.Text = "5.4";

this.textBox6.TextChanged += new System.EventHandler(this.textBox6\_TextChanged);

//

// label12

//

this.label12.AutoSize = true;

this.label12.Location = new System.Drawing.Point(335, 261);

this.label12.Name = "label12";

this.label12.Size = new System.Drawing.Size(23, 13);

this.label12.TabIndex = 19;

this.label12.Text = "mm";

//

// button3

//

this.button3.Location = new System.Drawing.Point(152, 287);

this.button3.Name = "button3";

this.button3.Size = new System.Drawing.Size(125, 23);

this.button3.TabIndex = 20;

this.button3.Text = "Cat am aruncat?";

this.button3.UseVisualStyleBackColor = true;

this.button3.Click += new System.EventHandler(this.button3\_Click);

//

// button4

//

this.button4.Location = new System.Drawing.Point(12, 316);

this.button4.Name = "button4";

this.button4.Size = new System.Drawing.Size(407, 23);

this.button4.TabIndex = 21;

this.button4.Text = "Ce greutate ar trebui sa folosesc ca firul sa fie intins la distanta aruncata de " +

"mine?";

this.button4.UseVisualStyleBackColor = true;

this.button4.Click += new System.EventHandler(this.button4\_Click);

//

// textBox7

//

this.textBox7.Location = new System.Drawing.Point(285, 352);

this.textBox7.Name = "textBox7";

this.textBox7.Size = new System.Drawing.Size(42, 20);

this.textBox7.TabIndex = 22;

this.textBox7.Text = "4";

this.textBox7.TextChanged += new System.EventHandler(this.textBox7\_TextChanged);

//

// textBox8

//

this.textBox8.Location = new System.Drawing.Point(297, 375);

this.textBox8.Name = "textBox8";

this.textBox8.Size = new System.Drawing.Size(45, 20);

this.textBox8.TabIndex = 23;

this.textBox8.Text = "45";

this.textBox8.TextChanged += new System.EventHandler(this.textBox8\_TextChanged);

//

// label13

//

this.label13.AutoSize = true;

this.label13.Location = new System.Drawing.Point(10, 352);

this.label13.Name = "label13";

this.label13.Size = new System.Drawing.Size(261, 13);

this.label13.TabIndex = 24;

this.label13.Text = "Care este inaltimea de la varful lansetei la planul apei?";

//

// label14

//

this.label14.AutoSize = true;

this.label14.Location = new System.Drawing.Point(333, 355);

this.label14.Name = "label14";

this.label14.Size = new System.Drawing.Size(15, 13);

this.label14.TabIndex = 25;

this.label14.Text = "m";

//

// label15

//

this.label15.AutoSize = true;

this.label15.Location = new System.Drawing.Point(10, 378);

this.label15.Name = "label15";

this.label15.Size = new System.Drawing.Size(281, 13);

this.label15.TabIndex = 26;

this.label15.Text = "Care este marimea unghiului format de varful lansetei si fir?";

//

// label16

//

this.label16.AutoSize = true;

this.label16.Location = new System.Drawing.Point(348, 378);

this.label16.Name = "label16";

this.label16.Size = new System.Drawing.Size(34, 13);

this.label16.TabIndex = 27;

this.label16.Text = "grade";

//

// button5

//

this.button5.Location = new System.Drawing.Point(36, 401);

this.button5.Name = "button5";

this.button5.Size = new System.Drawing.Size(293, 23);

this.button5.TabIndex = 28;

this.button5.Text = "La ce adancime se afla peste in momentul acesta?";

this.button5.UseVisualStyleBackColor = true;

this.button5.Click += new System.EventHandler(this.button5\_Click);

//

// button6

//

this.button6.Location = new System.Drawing.Point(77, 429);

this.button6.Name = "button6";

this.button6.Size = new System.Drawing.Size(232, 23);

this.button6.TabIndex = 29;

this.button6.Text = "Cat de aproape de suprafata este pestele?";

this.button6.UseVisualStyleBackColor = true;

this.button6.Click += new System.EventHandler(this.button6\_Click);

//

// Form1

//

this.AutoScaleDimensions = new System.Drawing.SizeF(6F, 13F);

this.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font;

this.ClientSize = new System.Drawing.Size(434, 464);

this.Controls.Add(this.button6);

this.Controls.Add(this.button5);

this.Controls.Add(this.label16);

this.Controls.Add(this.label15);

this.Controls.Add(this.label14);

this.Controls.Add(this.label13);

this.Controls.Add(this.textBox8);

this.Controls.Add(this.textBox7);

this.Controls.Add(this.button4);

this.Controls.Add(this.button3);

this.Controls.Add(this.label12);

this.Controls.Add(this.textBox6);

this.Controls.Add(this.label11);

this.Controls.Add(this.button2);

this.Controls.Add(this.label10);

this.Controls.Add(this.textBox5);

this.Controls.Add(this.label9);

this.Controls.Add(this.label8);

this.Controls.Add(this.button1);

this.Controls.Add(this.label7);

this.Controls.Add(this.label6);

this.Controls.Add(this.label5);

this.Controls.Add(this.textBox4);

this.Controls.Add(this.textBox3);

this.Controls.Add(this.textBox2);

this.Controls.Add(this.textBox1);

this.Controls.Add(this.label4);

this.Controls.Add(this.label3);

this.Controls.Add(this.label2);

this.Controls.Add(this.label1);

this.Name = "Form1";

this.Text = "Form1";

this.ResumeLayout(false);

this.PerformLayout();

}

#endregion

private System.Windows.Forms.Label label1;

private System.Windows.Forms.Label label2;

private System.Windows.Forms.Label label3;

private System.Windows.Forms.Label label4;

private System.Windows.Forms.TextBox textBox1;

private System.Windows.Forms.TextBox textBox2;

private System.Windows.Forms.TextBox textBox3;

private System.Windows.Forms.TextBox textBox4;

private System.Windows.Forms.Label label5;

private System.Windows.Forms.Label label6;

private System.Windows.Forms.Label label7;

private System.Windows.Forms.Button button1;

private System.Windows.Forms.Label label8;

private System.Windows.Forms.Label label9;

private System.Windows.Forms.TextBox textBox5;

private System.Windows.Forms.Label label10;

private System.Windows.Forms.Button button2;

private System.Windows.Forms.Label label11;

private System.Windows.Forms.TextBox textBox6;

private System.Windows.Forms.Label label12;

private System.Windows.Forms.Button button3;

private System.Windows.Forms.Button button4;

private System.Windows.Forms.TextBox textBox7;

private System.Windows.Forms.TextBox textBox8;

private System.Windows.Forms.Label label13;

private System.Windows.Forms.Label label14;

private System.Windows.Forms.Label label15;

private System.Windows.Forms.Label label16;

private System.Windows.Forms.Button button5;

private System.Windows.Forms.Button button6;

}

}

În cadrul acestei părți, cea de design a aplicației, există o componentă importantă și anume metoda InitializeComponent() în cadrul căreia sunt specificate pe rând toate elementele utilizate și totodată și dimensiunile lor pe ecranul aplicației, locul și forma acestora. Fiecare element este reprezentat prin nume, locație, mărime, index, text și o anumită funcționalitate care este scrisă în partea de cod a Formului.

Partea de cod a formului are următoarea formă:

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace FishingAplication

{

public partial class Form1 : Form

{

public double raza\_actiune\_tambur;

public double numar\_spire\_initiale;

public double dim\_spira\_initiala;

public double dim\_fir;

public double dim\_umpluta;

public double dim\_fir\_tambur;

public double dim\_u;

public double lungimea\_fir\_tambur\_plin;

public double lungimea\_fir\_tambur\_final;

public double lungime\_fir\_aruncat;

public double inaltime;

public double unghi;

public double adancime\_peste;

public Form1()

{

InitializeComponent();

}

private void button1\_Click(object sender, EventArgs e)

{

double copie\_pentru\_dim\_spira\_initiala = dim\_spira\_initiala;

dim\_fir\_tambur = 0;

double pas;

pas = dim\_fir \* 2 / 10;

while (raza\_actiune\_tambur > 0)

{

dim\_fir\_tambur += numar\_spire\_initiale \* copie\_pentru\_dim\_spira\_initiala / 100;

//consider ca numarul de spire ramane acelasi;

copie\_pentru\_dim\_spira\_initiala += pas;

raza\_actiune\_tambur -= dim\_fir;

}

MessageBox.Show("Pe mulineta dumneavoastra incape " + dim\_fir\_tambur+" m fir");

}

private void textBox1\_TextChanged(object sender, EventArgs e)

{

double nr = 0;

int i, dif;

string element = textBox1.Text;

int p = 1;

for (i = 0; i < element.Length; i++)

if (element[i] == '.' || element[i]==',')

{

dif = element.Length - i - 1;

while (dif > 0)

{

p = p \* 10;

dif = dif - 1;

}

}

else

if (element[i] >= '0' && element[i] <= '9')

nr = nr \* 10 + (int)Char.GetNumericValue(element[i]);

raza\_actiune\_tambur = nr / p;

}

private void textBox2\_TextChanged(object sender, EventArgs e)

{

double nr = 0;

int i, dif;

string element = textBox2.Text;

int p = 1;

for (i = 0; i < element.Length; i++)

if (element[i] == '.' || element[i] == ',')

{

dif = element.Length - i - 1;

while (dif > 0)

{

p = p \* 10;

dif = dif - 1;

}

}

else

if (element[i] >= '0' && element[i] <= '9')

nr = nr \* 10 + (int)Char.GetNumericValue(element[i]);

numar\_spire\_initiale = nr / p;

}

private void textBox3\_TextChanged(object sender, EventArgs e)

{

double nr = 0;

int i, dif;

string element = textBox3.Text;

int p = 1;

for (i = 0; i < element.Length; i++)

if (element[i] == '.' || element[i] == ',')

{

dif = element.Length - i - 1;

while (dif > 0)

{

p = p \* 10;

dif = dif - 1;

}

}

else

if (element[i] >= '0' && element[i] <= '9')

nr = nr \* 10 + (int)Char.GetNumericValue(element[i]);

dim\_spira\_initiala = nr / p;

}

private void textBox4\_TextChanged(object sender, EventArgs e)

{

double nr = 0;

int i, dif;

string element = textBox4.Text;

int p = 1;

for (i = 0; i < element.Length; i++)

if (element[i] == '.' || element[i] == ',')

{

dif = element.Length - i - 1;

while (dif > 0)

{

p = p \* 10;

dif = dif - 1;

}

}

else

if (element[i] >= '0' && element[i] <= '9')

nr = nr \* 10 + (int)Char.GetNumericValue(element[i]);

dim\_fir = nr / p;

}

private void textBox5\_TextChanged(object sender, EventArgs e)

{

double nr = 0;

int i, dif;

string element = textBox5.Text;

int p = 1;

for (i = 0; i < element.Length; i++)

if (element[i] == '.' || element[i] == ',')

{

dif = element.Length - i - 1;

while (dif > 0)

{

p = p \* 10;

dif = dif - 1;

}

}

else

if (element[i] >= '0' && element[i] <= '9')

nr = nr \* 10 + (int)Char.GetNumericValue(element[i]);

dim\_umpluta = nr / p;

}

private void button2\_Click(object sender, EventArgs e)

{

double copie\_pentru\_dim\_spira\_initiala = dim\_spira\_initiala;

double pas;

pas = dim\_fir \* 2 / 10;

lungimea\_fir\_tambur\_plin = 0;

while (dim\_umpluta > 0)

{

lungimea\_fir\_tambur\_plin = lungimea\_fir\_tambur\_plin + numar\_spire\_initiale \* copie\_pentru\_dim\_spira\_initiala / 100;

copie\_pentru\_dim\_spira\_initiala = copie\_pentru\_dim\_spira\_initiala + pas;

dim\_umpluta = dim\_umpluta - dim\_fir;

}

MessageBox.Show("Ati pus pe mulineta "+lungimea\_fir\_tambur\_plin+ " m fir din " + dim\_fir\_tambur+" posibili");

}

private void button3\_Click(object sender, EventArgs e)

{

double copie\_pentru\_dim\_spira\_initiala = dim\_spira\_initiala;

lungimea\_fir\_tambur\_final = 0;

double pas;

pas = dim\_fir \* 2 / 10;

while (dim\_u > 0)

{

lungimea\_fir\_tambur\_final = lungimea\_fir\_tambur\_final + numar\_spire\_initiale \* copie\_pentru\_dim\_spira\_initiala / 100;

copie\_pentru\_dim\_spira\_initiala = copie\_pentru\_dim\_spira\_initiala + pas;

dim\_u = dim\_u - dim\_fir;

}

lungime\_fir\_aruncat= lungimea\_fir\_tambur\_plin - lungimea\_fir\_tambur\_final;

MessageBox.Show("Felicitari! Ai aruncat "+lungime\_fir\_aruncat+" m fir" );

}

private void textBox6\_TextChanged(object sender, EventArgs e)

{

double nr = 0;

int i, dif;

string element = textBox6.Text;

int p = 1;

for (i = 0; i < element.Length; i++)

if (element[i] == '.' || element[i] == ',')

{

dif = element.Length - i - 1;

while (dif > 0)

{

p = p \* 10;

dif = dif - 1;

}

}

else

if (element[i] >= '0' && element[i] <= '9')

nr = nr \* 10 + (int)Char.GetNumericValue(element[i]);

dim\_u = nr / p;

}

private void button4\_Click(object sender, EventArgs e)

{

double distanta\_aruncata = lungime\_fir\_aruncat;

double grosime\_fir =dim\_fir;

double greutate\_metru\_fir = (Math.Pow((grosime\_fir / 2) , 2) \* 0.1) / (Math.Pow((0.155), 2));

double masa\_fir\_intins = 2 \* distanta\_aruncata \* greutate\_metru\_fir;

MessageBox.Show("Pentru a se intinde firul este nevoie de o greutate minima de: " + masa\_fir\_intins + " g");

}

private void textBox7\_TextChanged(object sender, EventArgs e)

{

double nr = 0;

int i, dif;

string element = textBox7.Text;

int p = 1;

for (i = 0; i < element.Length; i++)

if (element[i] == '.' || element[i] == ',')

{

dif = element.Length - i - 1;

while (dif > 0)

{

p = p \* 10;

dif = dif - 1;

}

}

else

if (element[i] >= '0' && element[i] <= '9')

nr = nr \* 10 + (int)Char.GetNumericValue(element[i]);

inaltime = nr / p;

}

private void textBox8\_TextChanged(object sender, EventArgs e)

{

double nr = 0;

int i;

string element = textBox8.Text;

for (i = 0; i < element.Length; i++)

if (element[i] >= '0' && element[i] <= '9')

nr = nr \* 10 + (int)Char.GetNumericValue(element[i]);

unghi = nr;

}

private void button5\_Click(object sender, EventArgs e)

{

double distanta\_aruncata = lungime\_fir\_aruncat;

adancime\_peste = distanta\_aruncata \* Math.Cos(unghi) - inaltime;

MessageBox.Show("Peste se afla la adancimea " + adancime\_peste + "m");

}

private void button6\_Click(object sender, EventArgs e)

{

double ip;

double distanta\_aruncata = lungime\_fir\_aruncat;

ip = (distanta\_aruncata \* adancime\_peste) / (inaltime + adancime\_peste);

MessageBox.Show("Peste a ajuns la "+ip+" m distanta de aer");

}

}

}

În această parte a aplicației se poate observa cum funcționează aceasta. Eu am utilizat o serie de textbox-uri în care utilizatorii aplicației vor adăuga, în funcție de labelul aflat înainte, care specifică o întrebare pentru aceștia, ce li se cere. În această parte de cod eu extrag ceea ce un utilizator va adăuga ca date de intrare în textboxul respectiv. Am utilizat și o serie de butoane care oferă răspunsul întrebărilor pe care le-am evidențiat în subcapitolul anterior. Pentru ca butoanele să ofere răspunsuri pentru intrebările respective am refolosit codul pe care l-am realizat în Matlab pentru modelele descrise anterior. Răspunsul pentru utilizator l-am oferit printr-un apel al funcției MessageBox.Show() care deschide o fereastra nouă in care acesta poate vedea rezultatul dorit.

**Concluzii**

După cum se poate constata pe baza chestiunilor tratate în lucrare, cunoștințele de specialitate căpătate în cei trei ani de studii în cadrul Facultății de Informatică pot fi folosite nu numai în domenii de importanță teoretică, ci și în domenii de utilitate practică imediată, chiar și cu tentă agrementară, cum ar fi pescuitul sportiv în cazul de față.

Realizarea acestor aplicații se va dovedi utilă atât celor ce doresc să-și cumpere o mulinetă cât și celor ce vor pescui la metoda “plantat”. În rândul pescarilor există foarte mulți, care nu se opresc la a dori să știe ce se întâmplă exact cu undița lor și vor cerceta mai mult, vor căuta soluții eficiente, se vor informa. În sprijinul acestora vin rezultatele din prezenta lucrare.

Pe parcursul dezvoltării lucrării de licență, au avut loc discuții cu o serie de pescari, cu diverși pasionați de pescuit, chiar și cu reprezentanți ai asociației de pescuit și vânătoare din Iași (AVPS). Fiecare dintre ei și-a manifestat interesul și dorința de a cunoaște și de a utiliza noile aplicații.

Pregătirea lucrării, documentarea pentru realizarea sa, au scos la iveală multiplele aplicabilități ale informaticii, legătura sa cu mediul înconjurător; s-au creat diverse conexiuni între domenii aparent disjuncte, s-au obținut rezultate interesante, rezultate care pot reprezenta un nou punct de plecare pentru alți pasionați de pescuit și de informatică.

Conchizând cele de mai sus, merită în principal evidențiat faptul că rolul unui informatician cu o pregătire adecvată în domeniu este cât se poate de important chiar și în direcții care, aparent, n-ar avea legatură cu informatica.

**Bibliografie**

1. Cartea “Pescuitul de la A la Z. Ghidul Pescarului Sportiv” . Editura Venus
2. <http://www.ghidul-scarului.ro/articoltot.php?articoltota&na=ISTORIA%20PESCUITULUI>
3. <http://www.crap.ro/pagina/60/247/sisteme-de-avertizare---avertizoare-electronice.html>
4. <http://shiva.pub.ro/PDF/TRA/L2_Introducere_in_Simulink.pdf>
5. Curs Matlab din cadrul Facultății de informatică Iași
6. <http://www.scritub.com/timp-liber/pescuit/SCURT-ISTORIC-AL-PESCUITULUI-S13615.php>
7. <http://pescuitamator.blogspot.ro/2013/06/tipuri-de-pescuit.html>
8. <https://www.mathworks.com/products/simulink.html>
9. <https://www.tutorialspoint.com/matlab/matlab_simulink.htm>
10. Istoria Pescuitului și a pisciculturii în Romania Vol.1 Constantin G. Giurescu, Buc 1964