TECHNICKÁ UNIVERZITA V LIBERCI
Fakulta zdravotnických studií

Převod fitness dat do tabulkového formátu

Bakalářská práce

Studijní program:	B3944 Biomedicínská technika
Studijní obor:	Biomedicínská technika
Autor práce:	**Jaroslav Váša**
Vedoucí práce:	doc. Ing. Josef Černohorský, Ph.D. Ústav mechatroniky a technické informatiky

Liberec 2020

TECHNICKÁ UNIVERZITA V LIBERCI
Fakulta zdravotnických studií

Zadání bakalářské práce

Převod fitness dat do tabulkového formátu

Jméno a příjmení: **Jaroslav Váša**
Osobní číslo: D17000039
Studijní program: B3944 Biomedicínská technika
Studijní obor: Biomedicínská technika
Zadávající katedra: Fakulta zdravotnických studií
Akademický rok: **2019/2020**

Zásady pro vypracování:

Konzultant: Ing. Pavel Jandura, Ph.D. a PhDr. Iva Šeflová, Ph.D.
Cíle práce:
1. Vytvoření programu na převod dat (GPS, výkon, tepová frekvence a další) pro uchování v tabulkovém formátu.
2. Data správně synchronizovat s daty získanými pomocí mobilní spirometrie.
3. Ověřit funkčnost vytvořeného programu.

Teoretická východiska (včetně výstupu z kvalifikační práce):
Stávající elektronika dovoluje záznam dat do vnitřní nevolatilní paměti a jejich export na cloud. Vlastní zpracování dat uživatelem je ale omezení, což brání dalšímu využití. Stávající softwarové vybavení, dovoluje export aktivit do formátu TCX, který je velmi podobný XML souboru, ale s fitness tagy, které je třeba upravit tak, aby je bylo možné snadno interpretovat v běžném tabulkovém editoru.
Výstupem bakalářské práce bude vytvořený program, který provede úpravu formátu TCX, tak aby bylo možné jeho interpretování v běžném tabulkovém editoru.

Výzkumné předpoklady / výzkumné otázky:
1) Předpokladáme, že vytvořený program zvýší efektivitu práce s již získanými daty
2) Předpokládáme, že program bude dále rozšiřován**.**

Metoda:
Kvantitativní.
Technika práce, vyhodnocení dat:
Analýza a měření.
Místo a čas realizace výzkumu:
Místo: Technická univerzita v Liberci, Fakulta mechatroniky, informatiky a mezioborových studií.
Čas výzkumu: prosinec 2019- únor 2020
Vzorek:
Funkčnost programu bude následně opakovaně ověřena. Počet ověření programu: 30-50.
Rozsah práce:
Rozsah bakalářské práce činí 50 – 70 stran (tzn. 1/3 teoretická část, 2/3 výzkumná část).

Rozsah grafických prací:
Rozsah pracovní zprávy:
Forma zpracování práce: tištěná/elektronická
Jazyk práce: Čeština

Seznam odborné literatury:

BARILLA, Jiří et al. 2016. *Microsoft Excel 2016: podrobná uživatelská příručka*. Brno: Computer Press. ISBN 978-80-251-4838-9.

BORY, Pavel. 2016. *C# bez předchozích znalostí*. Brno: Computer Press. ISBN 978-80-251-4686-6.

DIMON, Theodore. 2017. *Anatomie těla v pohybu: základní kurs anatomie kostí, svalů a kloubů*.Praha: Euromedia. ISBN 978-80-7549-158-9.

PASTUCHA, Dalibor. 2014. *Tělovýchovné lékařství: vybrané kapitoly.* Praha: Grada. ISBN 978-80-247-4837-5.

MELOUN, Milan et al. 2017. *Statistická analýza vícerozměrných dat v příkladech*. Praha: Univerzita Karlova, nakladatelství Karolinum. ISBN 978-80-246-3618-4.

ŠTOLL, Ivan et al. 2017. *Klasická teoretická fyzika*. Praha: Univerzita Karlova, nakladatelství Karolinum. ISBN 978-80-246-3545-3.

VRCHOVECKÁ, Pavlína. 2018. *Fyziologie člověka: učební texty. Liberec: Technická univerzita v Liberci*. ISBN 978-80-7494-418-5.

WAINEWRIGHT, Max. 2017. *Programuj: průvodce programováním krok za krokem*. Praha: Svojtka & Co. ISBN 978-80-256-2048-9.

HARTWELL, Jonathan. 2017. *C# and XML primer*. California: Apress. ISBN 978-1484225943 TROELSEN, Andrew a Philip JAPIKSE. 2017. *Pro C# 7: with .net and .net core.* New York: Springer Science+Business Media, ISBN 9781484230176.

BULAVA, Alan. 2017. *Kardiologie pro nelékařské zdravotnické obory*. Praha: Grada Publishing. ISBN 978-80-271-0468-0.

Vedoucí práce: doc. Ing. Josef Černohorský, Ph.D.
Ústav mechatroniky a technické informatiky

Datum zadání práce: 2. září 2019
Předpokládaný termín odevzdání: 30. června 2020

L.S.

prof. MUDr. Karel Cvachovec, CSc., MBA
děkan

V Liberci dne 31. ledna 2020

Prohlášení

Prohlašuji, že svou bakalářskou práci jsem vypracoval samostatně jako původní dílo s použitím uvedené literatury a na základě konzultací s vedoucím mé bakalářské práce a konzultantem.

Jsem si vědom toho, že na mou bakalářskou práci se plně vztahuje zákon č. 121/2000 Sb., o právu autorském, zejména § 60 – školní dílo.

Beru na vědomí, že Technická univerzita v Liberci nezasahuje do mých autorských práv užitím mé bakalářské práce pro vnitřní potřebu Technické univerzity v Liberci.

Užiji-li bakalářskou práci nebo poskytnu-li licenci k jejímu využití, jsem si vědom povinnosti informovat o této skutečnosti Technickou univerzitu v Liberci; v tomto případě má Technická univerzita v Liberci právo ode mne požadovat úhradu nákladů, které vynaložila na vytvoření díla, až do jejich skutečné výše.

Současně čestně prohlašuji, že text elektronické podoby práce vložený do IS/STAG se shoduje s textem tištěné podoby práce.

Beru na vědomí, že má bakalářská práce bude zveřejněna Technickou univerzitou v Liberci v souladu s § 47b zákona č. 111/1998 Sb., o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění pozdějších předpisů.

Jsem si vědom následků, které podle zákona o vysokých školách mohou vyplývat z porušení tohoto prohlášení.

19. dubna 2020 Jaroslav Váša

Poděkování

Rád bych poděkoval vedoucímu práce doc. Ing. Josefovi Černohorskému, Ph.D., za jeho neocenitelnou pomoc, rady a odborné vedení při tvorbě této bakalářské práce. Dále patří mé poděkování Ph.D. a PhDr. Ivě Šeflové, Ph.D., za užitečné informace a poskytnutá data z testování.

Anotace

Autor: Jaroslav Váša

Instituce: Technická univerzita v Liberci, fakulta zdravotnických studii

Název práce: Převod fitness dat do tabulkového formátu

Vedoucí práce: doc. Ing. Josef Černohorský, Ph.D.

Počet stran: 72

Rok obhajoby: 2020

Počet příloh: 1

Anotace: Tato bakalářská práce se zaměřuje na seznámení čtenáře s vlivem pohybové aktivity na lidský organismus. Dále také přibližuje důležitá spiroergonomická data a způsob měření těchto dat. Praktická část je zaměřena na vývoj programu schopného přeformátování fitness dat TCX do tabulkového formátu CSV, a zároveň tyto data správně synchronizovat s daty získanými pomocí mobilní spirometrie.

Klíčová slova: TCX, CSV, Spirometrie, Spiroergonomie, Sportovní medicína

Annotation

Author: Jaroslav Váša

Institution: Technical university of Liberec, Faculty of Health Studies

Title: Transfer of fitness data to spreadsheet format

Supervisor: doc. Ing. Josef Černohorský, Ph.D.

Pages: 72

Apendix: 1

Year: 2020

Annotation: This bachelor thesis focuses on acquainting the reader with the influence of physical activity on the human organism. It also discusses important spiroergonomic data and how to measure these data. The practical part is focused on the development of a program capable of reformatting TCX fitness data into a CSV spreadsheet format, and at the same time correctly synchronizing this data with the data obtained using mobile spirometry.

Keywords: TCX, CSV, Spirometry, Spiroergonomy, Sport medicine

Obsah

Seznam zkratek 11
1 Úvod 12
2 Teoretická část 13
2.1 Sportovní medicína 13
2.2 Pohyb v oblasti prevence 13
2.2.1 Vliv pohybové aktivity na lidský organismus 14
2.3 Tepová frekvence 16
2.3.1 Arteriální tep 16
2.3.2 Způsoby měření 16
2.4 Spirometrie 23
2.4.1 Statické plicní objemy 23
2.4.2 Dynamické plicní objemy 24
2.5 Spiroergometrie 25
2.5.1 Spiroergometrické ukazatele 25
3 Praktická část 33
3.1 Hardware využitý k měření 33
3.1.1 Zařízení Garmin 33
3.1.2 MetaMax 3B-R2 34
3.2 Popis formátu dat 36
3.2.1 TCX 36
3.2.2 CSV 38
3.2.3 XML data z mobilní spirometrie 39
3.3 Ošetření výjimek 40
3.3.1 Aktivní ošetření výjimek 41

3.3.2 Pasivní ošetření výjimek ... 42
3.4 Parsing ... 42
3.4.1 Serializace a deserializace XML ... 44
3.4.2 Parsování XML dat ... 45
3.5 Kódování znaků ... 45
3.6 Synchronizace ... 46
3.6.1 Převzorkování ... 46
3.6.2 GNSS a čas ... 48
3.6.3 Synchronizace cyklopočítače a mobilní spirometrie ... 50
3.6.4 Využitý algoritmus synchronizace ... 50
3.6.5 Další možnosti synchronizace ... 51
3.7 Popis programu ... 56
3.7.1 Blokové schéma programu ... 56
3.7.2 Využité metody ... 61
3.7.3 Grafické uživatelské rozhraní ... 63
3.7.4 Omezení vytvořeného programu ... 64
3.8 Návod k vytvořenému programu ... 65
4 Ověření funkčností programu ... 66
5 Závěr ... 67
Seznam použité literatury ... 68
Seznam příloh ... 72

Seznam zkratek

BTPS	Body temperature and pressure
CNS	Centrální nervová soustava
CSV	Comma separated values
ERV	Expirační reziduální objem
FEV_1	Jednovteřinová vitální kapacita
FVC	Poměr respirační výměny
GNSS	Globální družicový polohový systém
GPS	Globální polohový systém
GUI	Grafické uživatelské rozhraní
ID	Identifikátor
IRV	Inspirační reziduální objem
MEF	Maximální výdechové průtoky
PEF	Maximální výdechový proud vzduchu
PETCO2	Minutový příjem kyslíku
R	Poměr respirační výměny
RER	Poměr respirační výměny
RV	Reziduální objem
TCX	Training Center XML
VCO2	Minutový výdej oxidu uhličitého
VKP	Vitální kapacita plic
VO2 max	hodnota maximálního objemu kyslíku
VO2	Minutový příjem kyslíku
XML	Extensible Markup Language

1 Úvod

Stávající elektronika, využívaná pro snímání fitness dat, dovoluje záznam dat do vnitřní nevolatilní paměti a jejich export na cloud. Vlastní zpracování dat uživatelem je ale omezené, což brání dalšímu využití. Stávající softwarové vybavení, dovoluje export aktivit do formátu TCX, který je velmi podobný XML souboru, ale s fitness tagy, které je třeba upravit tak, aby je bylo možné snadno interpretovat v běžném tabulkovém editoru.

Právě tato bakalářská práce na téma Převod fitness dat do tabulkového formátu se si klade za cíl na výše zmíněnou problematiku převodu dat z formátů TCX do tabulkového formátu CSV. Druhým cílem práce je formátovaná data správně synchronizovat s daty, která byla získána během mobilní spirometrie. V rámci této práce byl vytvořen program, který převede TCX do CSV a při určitých omezeních zároveň tato data synchronizuje s daty z mobilní spirometrie. Poslední cíl práce byl ověřit vytvořený program jakožto výstup práce.

Teoretická část této práce přiblíží vliv pohybové aktivity na člověka, jaká data během takové aktivity lze sledovat, jaký význam tato data mají.

Praktická část začíná popisem hardwaru využitého k naměření hodnot, který má výstup práce přeformátovat. Zbytek praktické části je zaměřen na vývoj programu, který převede naměřená data TCX do tabulkového formátu CSV a zároveň tato data synchronizuje s daty z mobilní spirometrie. Praktická část obsahuje popis formátů, se kterými program pracuje, možnosti ošetření výjimek, metody parsování dat, možnosti synchronizace, popis algoritmu vytvořeného programu včetně blokového schématu, ověření funkčnosti programu.

V závěru jsou shrnuty a zhodnoceny výsledky práce.

2 Teoretická část

2.1 Sportovní medicína

Obor sportovní medicína, někdy také označován jako tělovýchovné lékařství, je interdisciplinární obor, který se zaměřuje na tělesnou aktivitu člověka ve spojení s lidským zdravím. Zaměřuje se na diagnostiku, léčbu i prevenci. Tento obor se nezaměřuje pouze na profesionální sportovce, ale také na běžnou populaci, kde je pohyb výraznou součásti všech částí prevence různých onemocnění. Ve sportovní medicíně se neřeší pouze klady tělesné aktivity, ale také negativa špatné kvality či kvantity zátěže, jež může vyústit v poškození zdraví. Jak již bylo výše zmíněno tento obor je interdisciplinární, protože pohyb se týká celého lidského organismu. Mezi obory, se kterými je dobrá úzká spolupráce v rámci sportovní medicíny patří vnitřní a všeobecné lékařství, kardiologie, rehabilitace, ortopedie, chirurgie, obezitologie, imunologie, pracovní lékařství atd. (Pastucha, 2014)

2.2 Pohyb v oblasti prevence

Pohybová aktivita má veliký význam v prevenci, od prevence primární po terciární. Primární prevence se zaměřuje na vše, co vede k předcházení nemoci. Tuto prevenci má na starosti čistě jedinec sám. Proto je v ní velice důležitý životní styl jedince, ať už jde o posilování, rozvíjení zdraví či stravu. (Šeflová, 2014)

Mezi primární a terciární prevenci se zařazuje prevence sekundární, na kterou má pohybová aktivita také vysoký vliv. Tato prevence nastupuje ve chvíli, kdy už nemoc probíhá, ale snažíme se omezit její progresi, nebo když je v časném stadiu a lze ji aplikovat k vyléčení dané nemoci. Na této fázi prevence se již podílí lékař. (Šeflová, 2014)

„Terciární prevence znamená předcházení opakování onemocnění a kvarterní prevence pak optimalizaci zbytkových funkcí a kvality života´´ (Šeflová, 2014, s.19). Jak už trochu z definice vyplývá, tato prevence je zaměřená na chronicky nemocné či starší jedince. U

těchto jedinců pomáhá pohyb hlavně psychicky. Fyzicky těmto osobám pomáhá udržet si stejnou či podobnou úroveň pohybových schopností. (Šeflová, 2014)

2.2.1 Vliv pohybové aktivity na lidský organismus

Pohybová aktivita má příznivé vlivy na lidský organismus. V příštích několika odstavcích bude nastíněno, jaké tyto vlivy jsou na některé systémy v lidském organismu.

Vliv pohybové aktivity na trávicí systém a metabolismus

Jednici, kteří mají pravidelnou pohybovou aktivitu vykazují zlepšení metabolismu lipidů. Další metabolismus, který se lepší je glukozový. Pozorujeme zvýšení citlivosti jednotlivých orgánů na insulin, který spolu s glukagonem reguluje hladinu cukrů v krvi. Pravidelný pohyb dále podporuje střevní peristaltiku a celý proces trávení. (Šeflová, 2014)

Pohybem můžeme předcházet civilizačním nemocem, jako jsou cukrovka a obezita. Obě zmíněné civilizační choroby vedou ke komplikacím, které mohou snížit kvalitu života. Civilizační choroby mohu za vyšší úmrtnost u jiných onemocnění jako je COVID-19, srdeční onemocnění či mrtvice.

Vliv pohybové aktivity na kostní a svalovou tkáň

Hlavní změny v oblasti kostní tkáně jsou ve změnách hustoty. Hustota kostní tkáně u lidí s aktivním životním stylem je v určitých částech těla vyšší, naopak u lidí s méně aktivním životním stylem si můžeme všimnout, že hustota kostní tkáně je v těchto oblastech nižší. Tento fenomén je následkem mechanického zatěžování, které vede k přestavbě kostních trámců a zvýšenému ukládání minerálních solí v mezibuněčných prostorách. Příznivé účinky pozorujeme i na šlachách a vazech. Jsou pevnější a více odolné vůči tahu. Nemoci, kterým lze pohybem v rámci kostní tkáně předcházet, jsou například osteoporóza, při které pohyb pomáhá i při již vzniklém onemocnění. (Šeflová, 2014)

Při pravidelné pohybové aktivitě dochází k hypertrofii, což znamená nárůst svalové hmoty. Další zlepšení ve svalové tkáni je ekonomizace svalové činnosti a její účinnosti a také zlepšení nervosvalové koordinace. (Šeflová, 2014)

S věkem samozřejmě ubývá svalová hmota a dochází i k řídnutí kostí., ale se správnou pravidelnou pohybovou aktivitou a správnou životosprávou, lze tyto změny zpomalit.

Vliv pohybové aktivity na kardiovaskulární systém

Vlivy pohybové aktivity jsou prokazatelné přímo na myokardu i na zbytku kardiovaskulárního systému. Pokud se člověk pravidelně věnuje pohybové aktivitě jeho myokard se stává výkonnější, více kontraktilní, zároveň zlepšuje svou ekonomiku práce a snižuje své nároky na kyslík i energii. Při vytrvalostním tréninku srdce reaguje zvýšením systolického a minutového objemu. Jak již bylo řečeno, zlepšení se netýká pouze myokardu, proto nesmíme opomenout zbytek kardiovaskulárního systému a jeho zlepšení. Všímáme si zlepšení mikrocirkulace, žilního návratu, snížení svalové perfuze (prokrvení). V lidském oběhu je díky pravidelné sportovní aktivitě také více krve, bez snížení hematokritu. (Šeflová, 2014)

Vliv pohybové aktivity na dýchací systém

Stejně tak jako u kardiovaskulárního systému si všímáme zlepšení v rámci hospodaření, v tomto případě u dýchání. Dýchací systém reaguje na pravidelnou pohybovou aktivitu zlepšením statických i dynamických faktorů nebo například zvýšením VO_{2max}. (Šeflová, 2014)

Více o dýchacím systému a jednotlivých parametrech v kapitole spiroergometrie.

Vliv pohybové aktivity na centrální nervový systém

Bylo dokázáno, že díky zvýšenému přívodu krve, tedy i kyslíku a živin do CNS vede ke změnám mikroskopickým i makroskopickým. Například zmnožení nervových spojů a

krevních vlásečnic. Dlouhodobě si můžeme všimnout, že pravidelná pohybová aktivita vede k větší odolnosti vůči stresu, ke zlepšení spánku, či zlepšení paměti. Pokud člověk zvolí intenzivnější zátěž, která by však měla trvat 30–60 minut, dočká se odměny ve formě pocitů štěstí, za které mohou endorfiny, které se při tomto typu zátěže začnou vyplavovat. (Vondruška a Dvořák, 1999)

2.3 Tepová frekvence

Tepová frekvence je jedna ze základních veličin, která vypovídá o současném stavu organismu. Tuto veličinu není nijak těžké měřit, proto je součástí mnoha komerčně dostupných zařízení na snímání pohybové aktivity. Více, jak o tepu tak o jeho snímaní přibližují následující kapitoly.

2.3.1 Arteriální tep

Arteriální tep je tlaková vlna, vyvolaná stahem srdce. Stah srdce vypudí objem krve do zbytku těla, postup tohoto objemu se projevuje jako tlaková vlna. Pulz je projevem funkce srdce a prokrvení tkání, proto je dobré ho monitorovat pro zjištění současného stavu organismu. U pulzu hodnotíme frekvenci, symetrii a rytmus. V případě frekvencí mluvíme o tachykardii (rychlejší frekvence) a bradykardii (pomalejší frekvence). Frekvence je ovlivněna několika faktory jako jsou věk, pohlaví, kondice, současný stav a vytížení organismu. Obecně můžeme říct, že v klidném stavu pacienta je tep 60–90 úderů za minutu v pořádku, pulz pod hranici 60 úderů za minutu je označován jako bradykardie a pulz nad 90 úderů za minut je označován jako tachykardie. Rytmus můžeme hodnotit jako pravidelný nebo nepravidelný. (Mourek, 2012)

2.3.2 Způsoby měření

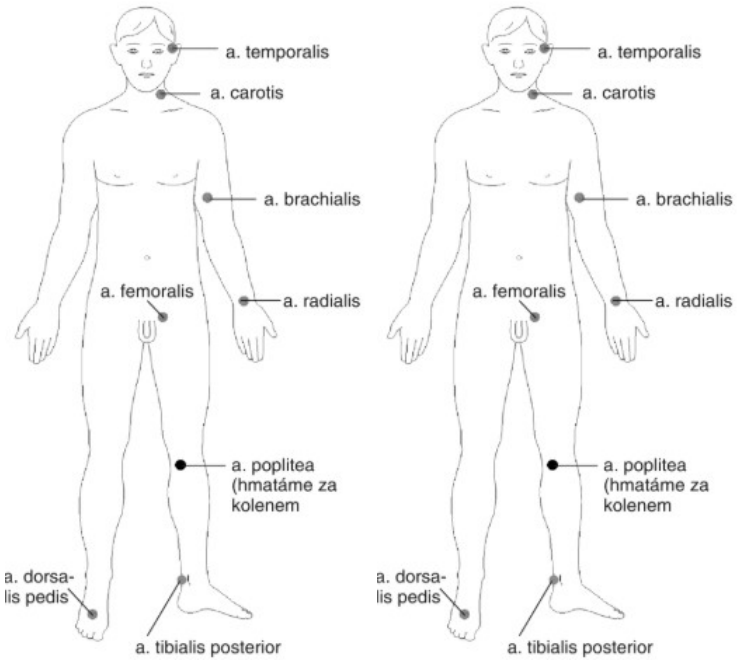
Arteriální tep můžeme zjišťovat a měřit dvěma způsoby. První metoda je klasické fyzikální vyšetření, při kterém člověk využívá svých vlastních smyslů. Do fyzikálního vyšetření patří:

- Vyšetření pohledem neboli adspekce
- Vyšetření pohmatem neboli palpace
- Vyšetření poslechem neboli auskultace

Druhý způsob vyšetření je pomocí přístroje, při kterém si při měření člověk zvolí vhodný přístroj pro naměření hodnot, popřípadě i vhodný software pro zpracování hodnot.

Palpační metoda

Při palpační metodě člověk využívá pouze svůj hmat, neboli své prsty, konkrétněji používáme 3 z 5 prstů, protože malíčku a palce během měření netřeba. Dva prsty využijeme k fixaci arterie, třetí prst je využit jako senzor a vyhodnocuje příchozí pulzní vlny. Arteriální tep lze nahmatat na několika arteriích, obecně platí, že dobře hmatatelné arterie jsou ty, které se nachází těsně pod povrchem těla. (Dobiáš, 2013) Na Obr. 1 Místa vyhmatání pulzu (Mikšová, 2006)Obr. 1 jsou vyobrazené arterie, na kterých lze dobře pulz nahmatat.



Obr. 1 Místa vyhmatání pulzu (Mikšová, 2006)

Přístrojové měření

Existuje mnoho přístrojů, které se využívají k měření a monitoraci tepové frekvence. Přístroj pro takové měření či monitoraci zvolíme podle podmínek při kterých měříme, dále také podle stavu pacienta. Při výběru je také třeba promyslet dostupnost přístroje. Po vybrání vhodného měřícího přístroje můžeme také přemýšlet nad stránkou softwaru, pokud očekáváme od přístroje samostatnou základní analýzu. Většina přístrojů je dobře připravena pro praxi v obou směrech. V následujících kapitolách budou přiblíženy metody měření pomocí Elektrokardiografie a optických metod.

Elektrokardiografie

Elektrokardiograf, zkráceně EKG, je přístroj, pomocí kterého jsme schopni snímat elektrickou aktivitu srdce. EKG nám vytvoří sumaci elektrického potenciálu všech buněk myokardu. K snímání se využívají elektrody, které dohromady vytváří svody. Výsledný produkt EKG je elektrokardiogram, na kterém je vyobrazená křivka elektrických potenciálů v závislosti na čase. Toto elektrofyziologické vyšetření srdce je neinvazivní a jednoduché, proto se těší velké oblibě v klinické praxi. V dnešní době je EKG základním vyšetření funkce srdce. (Bulíková, 2014)

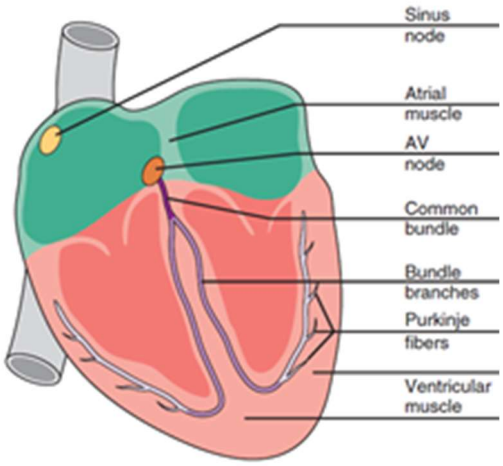
Převodní systém srdeční

Pro správné pochopení principu EKG je třeba znát převodní systém srdeční. V srdci nejsou pouze buňky umožňující mechanickou práci. Vedle těchto buněk je ještě specializovaná tkáň, ve které vznikají a převádí se elektrické vzruchy. Tato tkáň se nazývá převodní systém srdeční a má za úkol koordinovat činnost srdce. Pokud není převodní systém schopen zajistit správnou rytmickou aktivaci, vzniká různě závažné kardiologické onemocnění, podle toho, v jaké části, a jak přestal fungovat. (Bulíková, 2014) Níže jsou jednotlivé části převodního systému vypsány, včetně anglických překladu pro pochopení Obr. 2. Na Obr. 2 jsou tyto části ukázány na obrázku srdce.

Části převodního systému:

1. Sinoatriální uzel (zdroj vzruchů v srdci), Sinus node
2. Atrioventrikulární uzel (přechod vzruchu s předsíní na komory), AV node
3. Hisův svazek, Common bundle
4. Tawarova raménka, Bundle branches
5. Purkyňova vlákna, Purkinje fibers

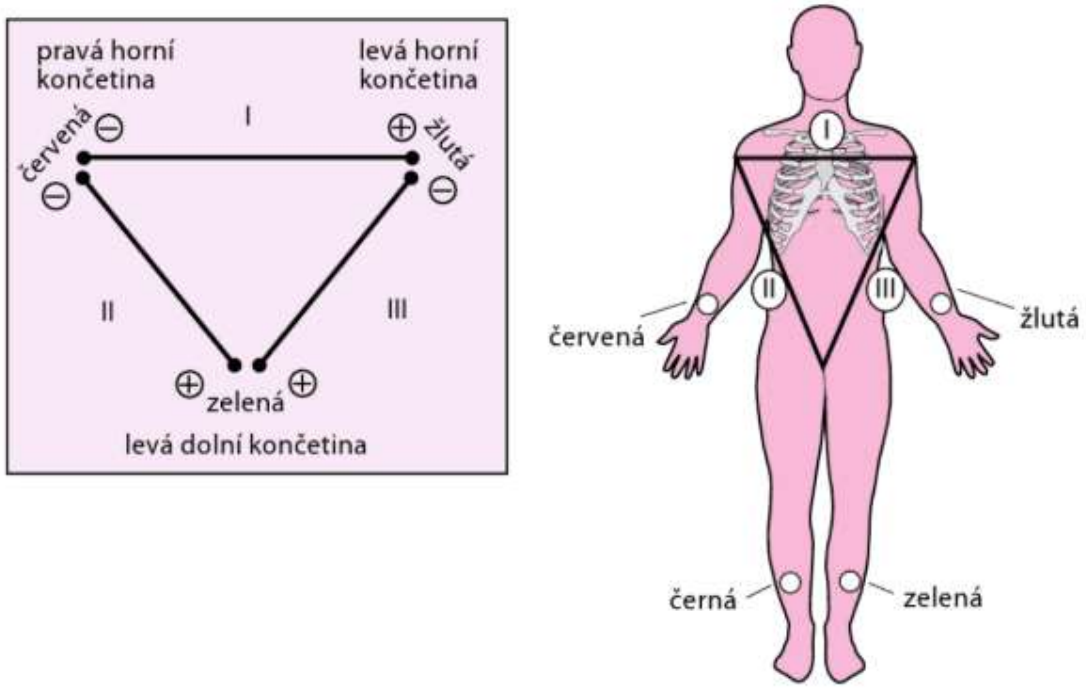
(Bulíková, 2014)



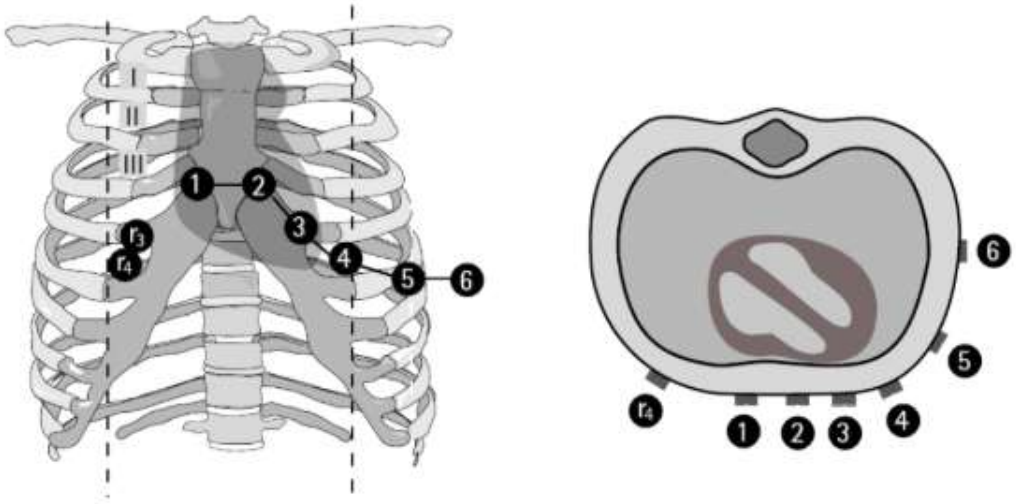
Obr. 2 Převodní systém srdeční (Ellenbogen et al., 2017)

EKG svody

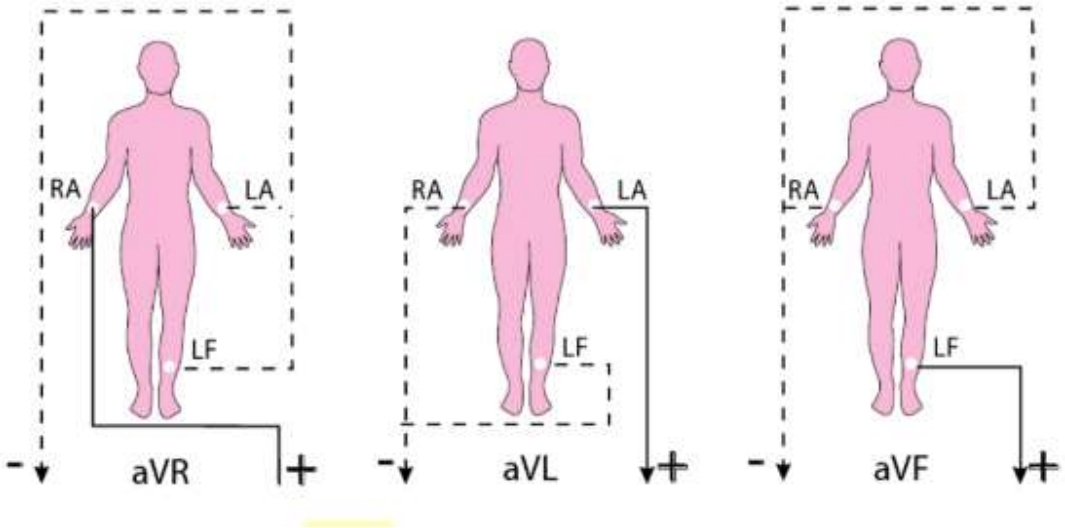
Pomocí svodů, vytvořených z elektrod, snímáme elektrický vektor popisující směr šíření a velikost elektrického potenciálu. Výsledný směr v okamžiku R kmitu nám určuje elektrickou osu srdeční, která je totožná s anatomickým uložením srdce v hrudníku. Ve výsledku máme u klasického EKG dvanáct svodů. Tři končetinové (Einthovenovy), které nám zachycují frontální rovinu viz Obr. 3. Kromě bipolárních Einthovenových máme ještě Goldbergovy končetinové svody (Obr. 5), které jsou unipolární a měří potenciál vůči indeferentní elektrodě, která je vytvořená spojením kabelů zbývajících dvou končetin. Šest hrudních svodů (Obr. 4), které zachycují výsledný vektor v horizontální rovině. (Sovová, 2006)



Obr. 3 Končetinové svody podle Einthovena (Haberl, 2012, s.13)



Obr. 4 Hrudní svody podle Wilsona (Haberl, 2012, s.14)



Obr. 5 Svody podle Goldberga (Haberl, 2012, s.15)

EKG princip

Když se vektor přibližuje ke kladné elektrodě zapíše přístroj kladnou výchylku, a naopak pokud se od kladné elektrody vzdaluje zapíše se výchylka negativní. Pro správné měření je důležitá správná příprava elektrod. Proto je důležité, správně připravit elektrody, to znamená správně nalepit nebo přisát a potřít EKG gelem, který pomáhá nejen k lepší vodivosti mezi povrchovou elektrodou a kůží pacienta, ale také přilnavosti, a uzemnit přistroj, aby nedošlo k rušení křivky. (Sovová, 2006) Čím více elektrod využijeme, tím lepší bude naše prostorové chápání postupu signálu převodním systémem.

EKG křivka a měření srdeční frekvence

Výsledná křivka se skládá z kmitů a vln, které jsou následně posuzovány. Posuzujeme kmity Q, R, S dohromady označovány jako QRS komplex. Vlny P, T, U a úseky PQ a ST. Amplituda jednotlivých vln a kmitu je v milivoltech. (Sovová, 2006)

Srdeční frekvenci jsme schopni měřit z EKG, protože kmity a vlny se pravidelně opakují podle frekvence srdeční aktivity. Nejpříhodnější je měřit kmit R, který má největší amplitudu. Měření je v dnešní době již v některých chytrých hodinkách, například Apple watch 4, stále se však nevyrovná klasickému dvanácti svodovému EKG. A rozhodně by

nemělo klasické měření nahrazovat, ale může upozornit na problémy, kterých by si normálně člověk nevšiml, aby vyhledal odbornou pomoc.

Pletysmografie

Pletysmografie slouží ke zjištění změn objemu. Lze měřit jak změny v celém těle, tak pouze v části. Toto vyšetření je neinvazivní. Pomocí této metody jsme schopni měřit změny způsobené změnou prokrvení, ale také změny vyvolané dýcháním. Všechny měřené objemové změny jsou zaznamenávány v závislosti na čase a vytváří se tak pletysmografická křivka. Je několik fyzikálních principů, které lze využít v pletysmografii. Například kapacitní, impedanční nebo piezoelektrický, v této práci se zaměříme na fotoelektrický, který je součástí moderních sporttesterů. (Kolář, 2007)

Fotoelektrický pletysmograf

Tento přístroj využívá vlastností tkáně při průchodu světla, kdy dochází k jeho odrazu, absorpci nebo rozptylu. Tyto vlastnosti se mění při změně objemu krve v tkáni při systole. Máme dvě možnosti, jak tyto změny snímat. Reflexní metodou nebo průsvitovou (tranmisní). Rozdíl je v tom, že reflexní má snímač i zdroj paprsků na stejné straně tkáně a průsvitný pletysmograf má zdroj na jedné a snímač na druhé straně měřené tkáně. Reflexní tedy snímá paprsky světla, které jsou odražené na rozdíl od průsvitového, který snímá paprsky průchozí. Jako zdroj světla se používá infračervená dioda, která používá vlnové délky nad 800 nm, aby nedošlo ke zkreslení měření, kvůli rozdílu absorpce světla okysličené a neokysličené krve. Na této vlnové délce je rozdíl absorpce světla u krve pouze malý. (Kolář, 2007)

Sporttestery využívají často metodu reflexní a umisťují snímač i zdroj například na zadní stranu hodinek a měření probíhá v oblasti zápěstí.

2.4 Spirometrie

Patří mezi funkční vyšetření plic, které funguje jako funkční pulmonální test. Spirometrie nám pomáhá získat informace o částech zevního plicního dýchání. Tyto části zevního plicního dýchání rozdělujeme na výměnu, distribuci, difuzi plynů a prokrvení plicních kapilár. Spirometrie vytváří grafy popisující závislost průtoku na objemu nebo objemu na čase. Spirometrie se stala díky parametrům, které je schopna měřit, jednou ze základních metod diagnózy obstrukčních, restrukčních a dalších plicních poruch. (David,2008)

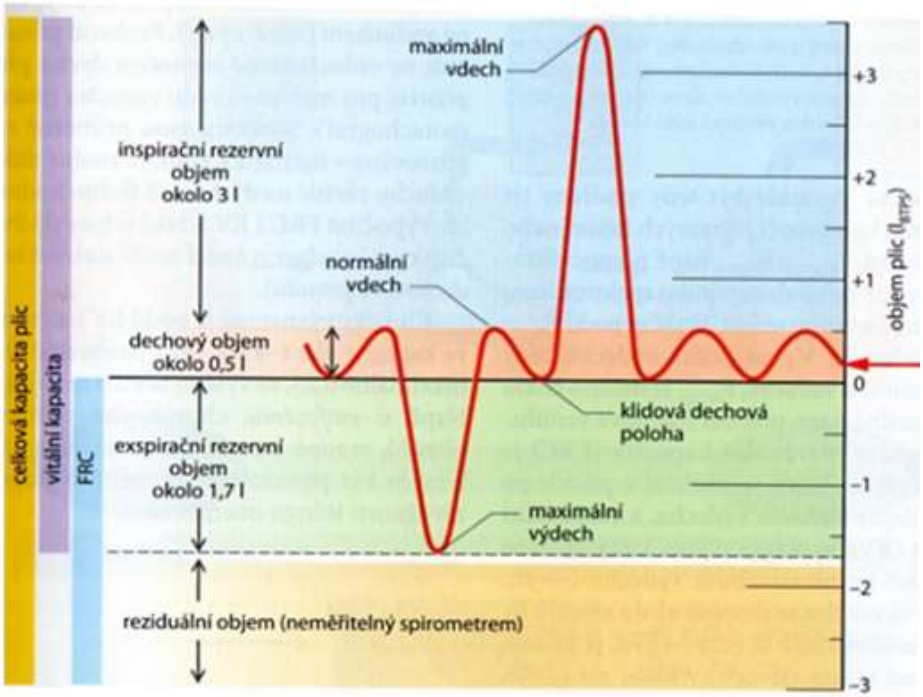
2.4.1 Statické plicní objemy

V plicích rozlišujeme několik objemů. Každý z těchto objemů má důležitou fyziologickou funkci. Velikost plicních objemů poukazuje na správnou funkci plic. (Mourek, 2012)

Pokud nás zajímá množství vydýchaného vzduchu, můžeme spočítat minutovou ventilaci. Běžný objem vydýchaného vzduchu za minutu se pohybuje okolo 7,5 litrů, tento objem je klidový, při zátěži se zvětšuje až na 150 l/min, někdy až na 170 l/min. Vypočítává se násobením klidového dechového objemu a dechové frekvence. Klidový dechový objem by měl být 500 ml. Tento objem se rozděluje na vzduch, který se nachází v mrtvém dýchacím prostoru (150 ml) a alveolární vzduch (350 ml). (Mourek, 2012)

Další plicní objem se nazývá expirační rezervní objem (ERV). ERV je objem, který lze maximálně vydechnout po předcházejícím klidovém výdechu. Pohybuje se okolo 1,1 litru. Opakem ERV je inspirační rezervní objem (IRV). Jak již název napovídá, jde o maximální nádech po klidném nádechu, bývá okolo 2 až 3 litrů. Vitální kapacita plic (VKP), je součet ERV a IRV. VKP je velmi závislá na faktorech jako jsou věk, pohlaví, hmotnost, ale také životní styl. Hodnoty VKP jsou většinou v rozmezí 3–5 litrů. (Mourek, 2012)

Poslední objem je reziduální (RV), který se pohybuje okolo 1,2 litrů. Je to objem, který zůstává v plicích i po maximálním výdechu a je součtem kolapsového a minimálního objemu. Tento objem nejsme schopni měřit pomocí spirometrie. Kolapsový znamená objem, který se uvolní pneumothoraxem a minimální je objem, který zůstává i po pneumothoraxu. (Mourek, 2012) Všechny plicní objemy jsou znázorněny na Obr.6.



Obr.6 Záznam plicního objemu v závislosti na čase, na kterém jsou znázorněny jednotlivé plicní objemy a kapacity. (Zdroj: Silbernagl,Atlas fyziologie člověka str. 113)

2.4.2 Dynamické plicní objemy

Všechny výše uvedené plicní objemy jsou statické. Níže uvedené plicní objemy označujeme jako dynamické, protože jsou popisované křivkou objemu v závislosti na čase. (Mourek, 2012)

Usilovná vitální kapacita (FVC – Forced Vital Capcity) popisuje objem vzduchu, který je pacient schopen vydechnout při maximálním nádechu a úsilí za jednotku času. (David,2008)

Jednosekundová vitální kapacita (FEV_1) je část FVC v první sekundě. K měření se používá Tiffeneauv test. Výsledky se udávají v procentech FVC nebo v objemových jednotkách. (David,2008)

Vrcholový výdechový průtok (PEF) je maximální průtok během vynuceného výdechu. V přímé souvislosti jsou zde i maximální výdechové průtoky (MEF), které se měří jen během části vydechovaného objemu. (David,2008)

Střední nádechový průtok (MIF50) se měří na úrovni 50% nadechnutého FVC. Vrcholový nádechový průtok řeší maximální průtok během nádechu. (David,2008)

2.5 Spiroergometrie

Spiroergometrie je vyšetření zabývající se reakcí organismu, v rámci plicní ventilace a výměny plicních plynů při zátěži. V dnešní době se používá pro posouzení prognózy a indikace k transplantaci srdce u pacientů, kteří mají srdeční selhání. (Chaloupka, 2003)

2.5.1 Spiroergometrické ukazatele

Spiroergonomické ukazatele, které se měří a co znamenají vysvětlí následující odstavce.

Minutová ventilace (V)

Minutová ventilace se udává v litrech za minutu, jde tedy o objem vydechovaného vzduchu během jedné minuty. Pro dospělé osoby v klidu se udávají fyziologické hodnoty 5 – 6 $l.min^{-1}$. Minutová ventilace je silně závislá na míře zátěže. To je projev zvýšených nároků na aerobní získávání energie, ale také na odvedení oxidu uhličitého z těla, aby nedošlo k zátěžové acidóze. U mužů v dospělosti dosahuje hodnota minutové ventilace až k 200 $l.min^{-1}$ během maximální zátěže. Po skončení zátěže ventilace klesá zpátky ke klidovým hodnotám, tento proces trvá zhruba 10–15 minut. (Novotný, 2017)

Ventilační práh

Pokud se podíváme na křivku závislosti minutové ventilace na tělesné zátěži, všimneme si, že v určitou chvíli nastává zlom křivky. Křivka začne strmě stoupat. Tento zlom nazýváme ventilační práh a je způsobený potřebou organismu vydechovat více oxidu uhličitého. (Novotný, 2017)

Minutový příjem kyslíku (VO2)

Minutový příjem kyslíku nám říká, jak velký je objem kyslíku za jednu minutu. Udává se stejně jako minutová ventilace v $l.min^{-1}$ nebo v $ml.min^{-1}$. Pro lepší zhodnocení výsledků se výsledek ještě dělí váhou testovaného subjektu v kilogramech a vynásobí faktorem BTPS (Body temperature and pressure), ten nám zajišťuje zohlednění aktuální atmosférické situace. Pro jedince, který je v klidu, se udává VO2 3,5 $ml.min^{-1}$. Zvýšení tohoto ukazatele v klidu může být známkou potřeby aerobní regenerace tkání, ale také závažnějších věcí jako je například zánět či hypertyreóza. VO2 je lineárně závislá na intenzitě zátěže. Když se náhle zvýší, poukazuje na nedostatečnou regeneraci sil, přetrénování, ale také možné onemocnění. (Novotný, 2017)

Další ukazatel, který přímo navazuje na VO2 je VO2max. Jak už název napovídá jde o maximální VO2, kterého je jedinec schopen dosáhnout. Je to v současné době nejlepší ukazatel kapacity transportního systému pro kyslík, a tedy i míry schopnosti získávat energii pro svaly, které se účastní pohybu. (Novotný, 2017)

VO2max lze interpretovat v procentech pro lepší srovnání osob s rozdílnými hodnotami VO2max. Vzorec pro výpočet procentuálního VO2max popisuje rovnice (1). Kde VO2z vyjadřuje minutový příjem kyslíku při zátěži a VO2k během klidu. (Novotný, 2017)

$$VO2max\% = ((VO2z - VO2k) / (VO2max - VO2k)) * 100 \quad (1)$$ (Novotný, 2017)

Kyslíkový dluh

Kyslíkový dluh je projevem anaerobního získávání energie pro pracující svaly. Vypočítá se z objemu kyslíku přijatého organismem ihned po zátěži, po odečtení klidového VO2. Tento dluh se nazývá čistý kyslíkový dluh. Vznik kyslíkového dluhu je způsoben aerobní regenerací energetického metabolismu, jako je doplnění glukosy a glykogenu. Doba splácení kyslíkového dluhu je závislá na míře zátěže. (Novotný, 2017)

Kromě klasického kyslíkového dluhu rozlišujeme ještě maximální kyslíkový dluh, který je ukazatelem kapacity anaerobního energetického metabolismu. (Novotný, 2017)

Tepový kyslík

Počítá se podílem přijmu kyslíku a minutové srdeční frekvence. Pomocí tohoto výpočtu jsme schopni odhadnout kolik objemu kyslíku je vypuzeno během srdeční systoly do oběhu. Tento nepřímý ukazatel funkční kapacity myokardu může upozornit na přetížení myokardu, pokud se při normální zátěži objevují nižší hodnoty. (Novotný, 2017)

Minutový výdej oxidu uhličitého (VCO2)

Popisuje výdej oxidu uhličitého organismem během jedné minuty. U zdravých jedinců by měl být mírně nižší než VO2. (Novotný, 2017)

Poměr respirační výměny (R, RER)

Poměr VCO2 a VO2, je ukazatelem výměny plynů ve svalové tkáni. Pokud je člověk v klidu, pohybuje se okolo 0,70. Na začátku zátěže stoupá. Pokud není zátěž lehká, v tu chvíli pozorujeme ze začátku mírný pokles. Na konci stupňované zátěže dosahuje maxima. Po zátěži se pomalu vrací ke klidovým hodnotám. (Novotný, 2017)

Anaerobní práh (LA, V-R)

Anaerobní práh popisuje, kdy se v lidském organismu mění převážně aerobní energetický metabolismus na anaerobní. Označuje časový úsek během stupňované zátěže, kdy se nejprve zvýší anaerobní krytí energie, tím se zvyšuje krevní laktát. Krevní laktát snižuje pH a množství HCO3-, to vede k zvýšené dechové aktivitě, která zvýší minutovou ventilaci CO2, R a V/VO2. Parciální tlak oxidu uhličitého (PETCO2) při výdechu je konstantní, označujeme jej jako izokapnickou kompenzaci metabolické acidózy. Po této fázi, pokud stále zvyšujeme zátěž, se začne zvyšovat V/VCO2 a PETCO2 se snižuje. Tuto kompenzaci metabolické acidózy označujeme jako respirační. (Novotný, 2017)

Důvody stanovení anaerobního prahu

Díky anaerobnímu prahu, jsme schopni vyjádřit, jak se při rozvíjející se anaerobní glykolýze během narůstající zátěže mění respirační a metabolické ukazatele (Novotný, 2017). V dalších několika odstavcích budou popsány přesnější důvody pro stanovení anaerobního prahu.

Hodnocením aerobní schopnosti kyslíkového transportního systému můžeme predikovat poškození tohoto systému. Nepoužívá se pouze u sportovců, ale také u kardiologických pacientů. (Novotný, 2017)

Funguje i jako hodnocení připravenosti a účinnosti vytrvalostního (aerobního) tréninku sportovce. Připravenost predikujeme pomocí rychlosti při anaerobním prahu. Účinnost poznáme tak, že po efektivním tréninku sledujeme posunutí anaerobního prahu do vyšších intenzit. (Novotný, 2017)

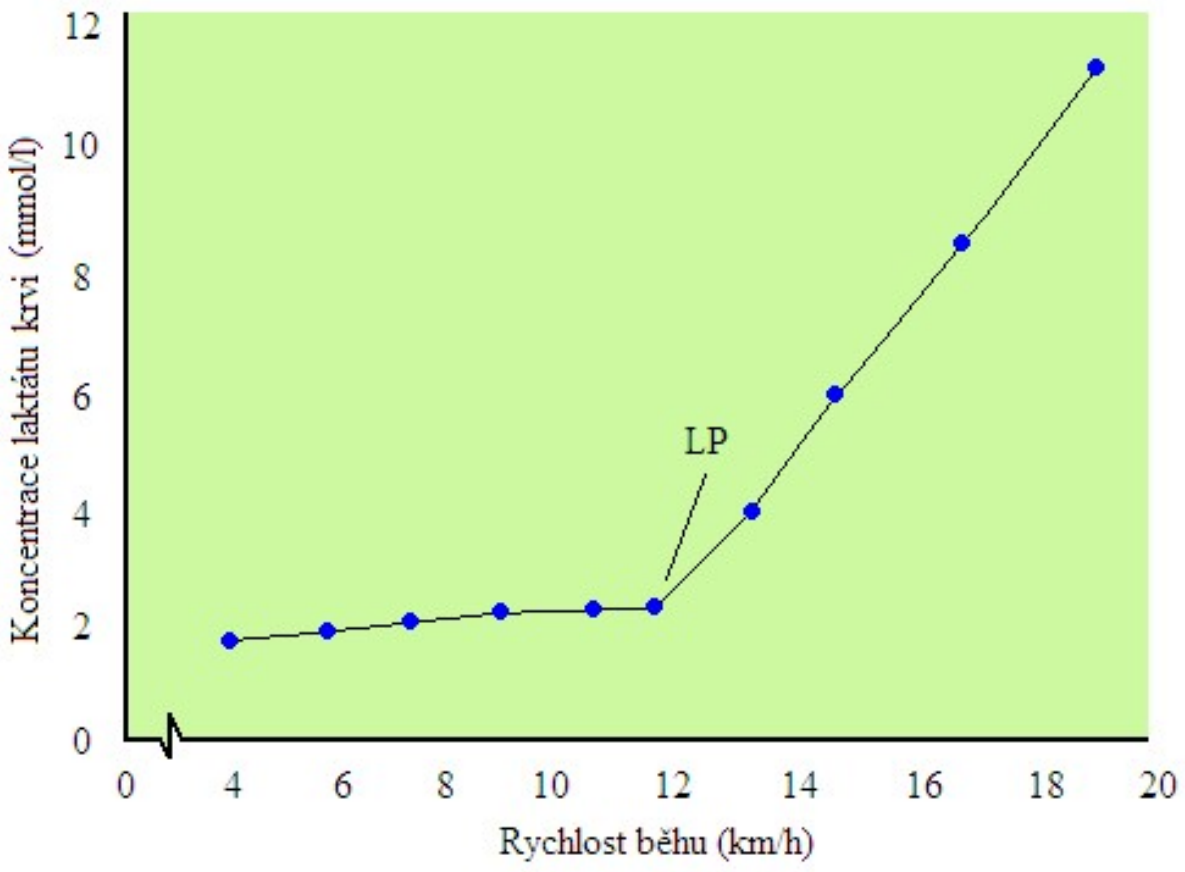
Pomáhá s plánovaním aerobního i anaerobního tréninku. Rozdělení do čtyř tréninkových pásem od lehkého běhu po rychlý běh při aerobním intervalovém tréninku. (Novotný, 2017)

Určení anaerobního prahu

K určení anaerobního prahu využíváme tři metody. Těmito metodami jsou určení z laktátového, ventilačního nebo cirkulačního prahu. Následující části popisují každý z těchto prahů.

Laktátový práh

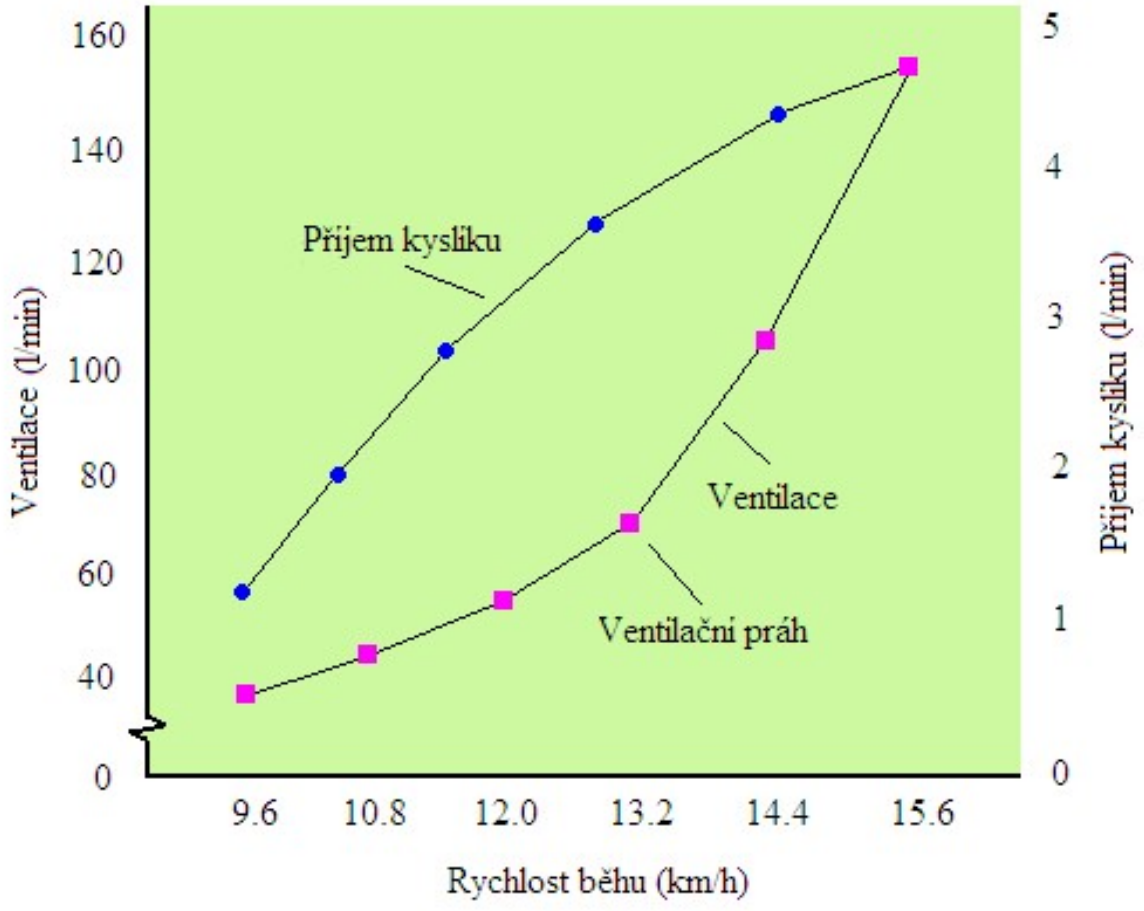
První možnost určení anaerobního prahu je pomocí laktátové křivky, kterou vytváříme během stupňující se zátěže. Laktátová křivka je vyobrazena na Obr.7. Odebíráme vzorky krve, ze které se stanovuje koncentrace laktátu v krvi a postupně vytváříme křivku. V místě zlomu se začal laktát hromadit ve větším množství. Tento zlom nazýváme laktátový práh. Nachází se mezi 2-8 mmol/l. (Bernaciková, 2012)



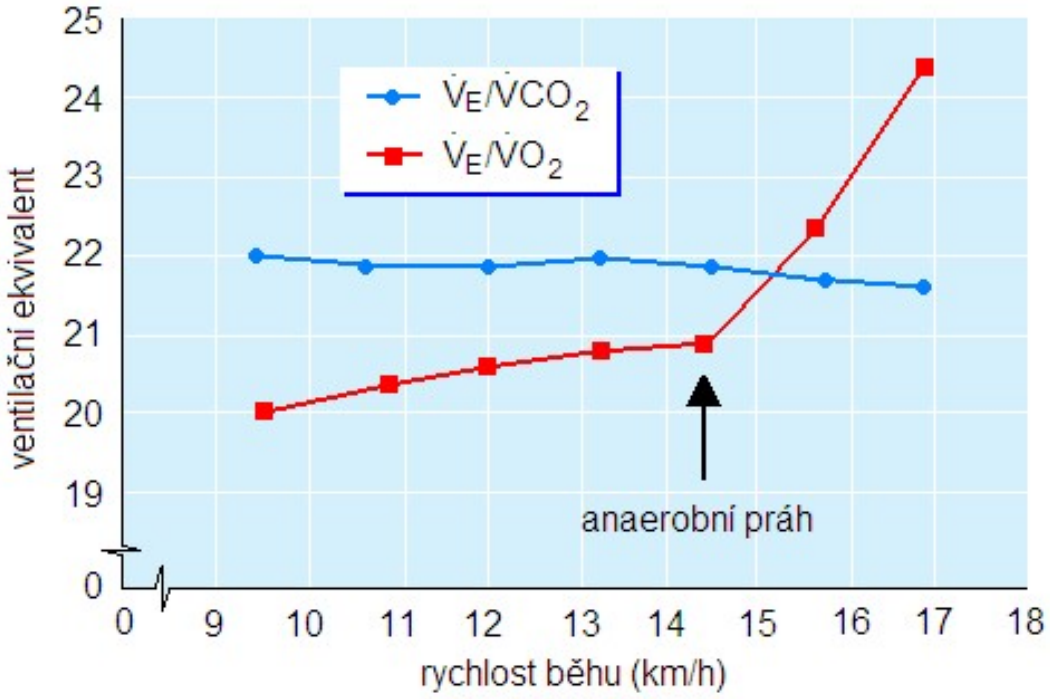
Obr.7 Laktátový práh (Zdroj: Bernaciková, 2012)

Ventilační práh

Další možnost zjištění anaerobního prahu je z parametrů ventilačních. První parametr, ze kterého lze zjistit anaerobní práh, je ventilační křivka, popsána obrázkem Obr.8. Druhá možnost je z ventilačního ekvivalentu pro kyslík, popsáno Obr.9. Poslední z uvedených možností je stanovení prahu z poměru respirační výměny, kde hledáme, kdy VCO2 je rovno VO2. (Bernaciková, 2012)



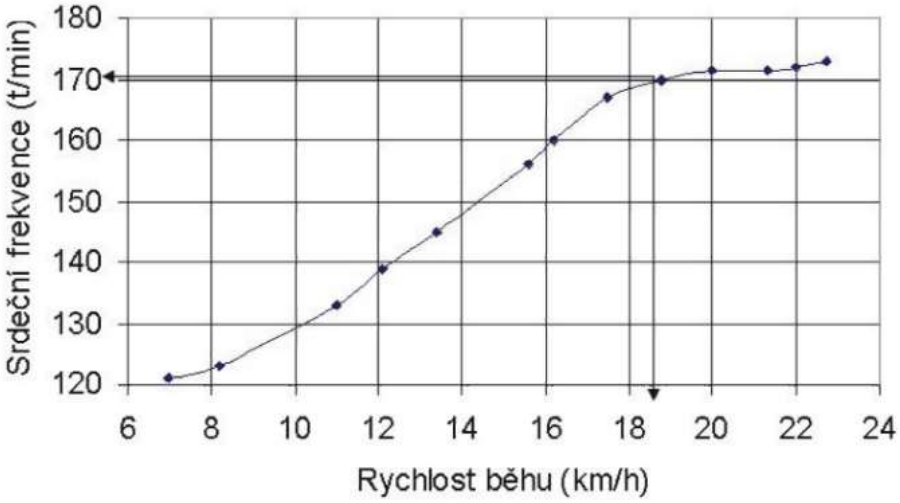
Obr.8 Ventilační práh (Zdroj: Bernaciková,2012)



Obr.9 Anaerobní práh z ventilačního ekvivalentu pro kyslík (Zdroj: Bernaciková)

Kterou metodu k nalezení ventilačního prahu využijeme je na nás.

Cirkulační práh



Obr. 10 Příklad stanovení „cirkulačního prahu“ (Novotný,2006)

V této metodě hledání aerobního prahu se využívá Conconiho testu. Nevýhoda této metody je, že je nepřímá, a proto není tak přesná, jako například měření laktátu v krvi.

Vychází z odhadu prahu z křivky vytvořené ze závislosti srdeční frekvence na rychlosti běhu. Na této křivce hledáme rychlost běhu, kdy křivka přestává být lineární. (Bernaciková, 2012). Křivka viz Obr. 10.

3 Praktická část

Celá programová část práce byla vypracována v programovacím jazyce C#, proto všechny příklady, jako je ošetření výjimek, kódování znaků, parsování a příklady přímo z vyhotoveného programu, jsou tvořeny v jazyce C#.

3.1 Hardware využitý k měření

Předtím než se s daty pracuje musejí být naměřena. K měření TCX dat byl využit cyklopočítač Garmin edge 520 a wattmetry Garmin vector. K naměření dat z mobilní spirometrie byl využit přístroj MetaMax 3B-R2.

3.1.1 Zařízení Garmin

V této práci bylo využito dvou zařízení od společnosti Garmin. Jaká zařízení to byla a co nabízejí bude popsáno několika dalších odstavcích

Garmin edge 520 je cyklopočítač vyvinutý společností Garmin. Tento počítač byl, ve spolupráci s pedálovými wattmetry, využit v této práci k zaznamenávání TCX dat.

Díky zabudované ANT+ komunikaci není problém připojit k zařízení snímač srdečního tepu, sensor rychlosti či wattmetr. Samozřejmě nechybí ani bluetooth komunikace s telefony s operačním systémem Android nebo IOS. To lze využít k funkci LiveTracking, která se používá k živému přenosu pozice cyklisty. (Garmin, 2018)

O podrobné zobrazení a analýzu naměřených hodnot se stará on-line tréninkový deník Garmin Connect. Data z tohoto deníku můžeme exportovat například do TCX. (Garmin, 2018)

Garmin vector 3 jsou cyklistické pedálové wattmetry vyvinuté společností Garmin. Díky umístění wattmetrů do pedálů má spoustu výhod oproti klasickému umístění v náboji zadního kola. Mezi tyto výhody patří lehká přenositelnost, minimum ztráty dat, ale také možnost měření na každém pedálu zvlášť, a tím určit procentuální výkon pravé či levé

nohy. (Garmin, 2018) Aby wattmetry správně fungovaly, je třeba využít tretry se systémem LOOK.

Princip měření výkonu je založen na výchylce osy pedálu během šlapání. Toto měření probíhá během celého šlapání a je posíláno bezdrátovou komunikací ANT+ do cyklopočítače, kde jsou data zaznamenávána s ostatními informacemi, jako jsou čas, vzdálenost, tep atd. (Garmin, 2018)

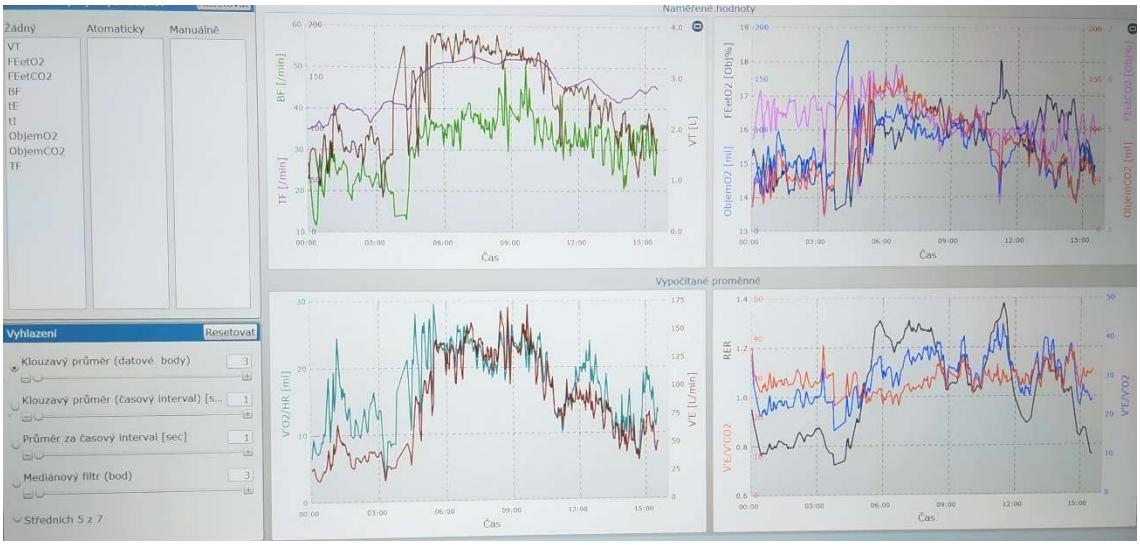
V případě této práce byl využit Garmin vector 2, kvůli ekonomické dostupnosti. Rozdíl mezi v vectorem 2 a 3 je v zabudování na kolo. Vector 2 lze spojit s velkou řadou Edge zařízení.

3.1.2 MetaMax 3B-R2

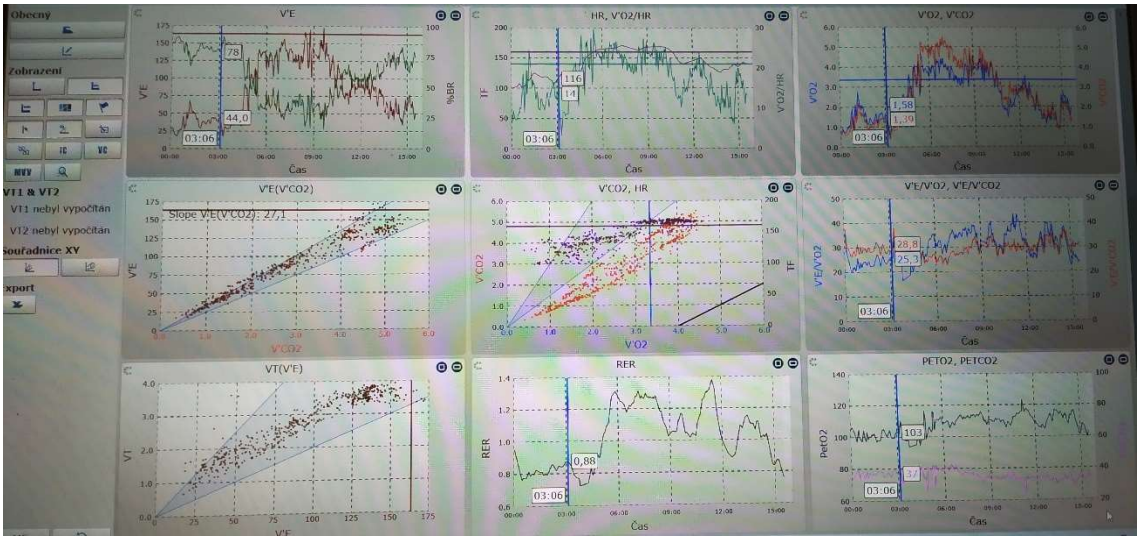
MetaMax 3B-R2 je systém využívaný pro zátěžovou spirometrii. Je ideální pro zátěžové testy mimo laboratoř. Po připojení rozšiřujících softwarových a hardwarových prvků je přístroj schopný fungovat jako mobilní laboratoř. (Compek, 2010)

Tento přístroj byl využit k měření dat z mobilní spirometrie hodnot, se kterými se dále v této práci pracovalo. Jak vypadá MetaMax 3B-R2 lze vidět na Obr. 11. Software, který data interpretuje se jmenuje Cortex MetaSoft Studio, vyobrazen na Obr. 12 a Obr. 13 .

Obr. 11 MetaMax 3B-R2 (Zdroj: Vedoucí práce)



Obr. 12 Časové průběhy spirometrických veličin z prostředí Metasoft studio část 1. (Zdroj: Vedoucí práce)



Obr. 13 Časové průběhy spirometrických veličin z prostředí Metasoft studio část 2. (Zdroj: Vedoucí práce)

3.2 Popis formátu dat

3.2.1 TCX

Formát TCX (Training Center XML) je vytvořený společností Garmin. TCX je formát založený na XML (eXtensible Markup Language) a slouží k ukládání a posílání dat mezi některými produkty od společnosti Garmin. Data, která se ukládají v TCX jsou GNSS a fitness data.

Jak už bylo výše zmíněno TCX vychází z XML, a proto i jeho syntaxe je velmi podobná. Stejně jako XML využívá elementů, tak TCX využívá fitness tagů. Název tagu popisuje, o jakou hodnotu se jedná. Syntaxe je tedy následující: <Fitness tag> hodnota </Fitness tag>.

V horní části dokumentu je vždy napsáno, o jakou verzi XML se jedná a jaké se používá kódování řetězců, jinak řečeno, jakou sadu znaku používáme. Před záznamem aktivit máme ještě odkazy na schémata, která se v dokumentu používají Obr. 14.

```
<?xml version="1.0" encoding="UTF-8"?>
<TrainingCenterDatabase
  xsi:schemaLocation="http://www.garmin.com/xmlschemas/TrainingCenterDatabase/v2 http://www.garmin.com/xmlschemas/TrainingCenterDatabasev2.xsd"
  xmlns:ns5="http://www.garmin.com/xmlschemas/ActivityGoals/v1"
  xmlns:ns3="http://www.garmin.com/xmlschemas/ActivityExtension/v2"
  xmlns:ns2="http://www.garmin.com/xmlschemas/UserProfile/v2"
  xmlns="http://www.garmin.com/xmlschemas/TrainingCenterDatabase/v2"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:ns4="http://www.garmin.com/xmlschemas/ProfileExtension/v1">
```

Obr. 14 TCX data část 1. (Zdroj: Autor)

Následující tagy se již týkají přímo aktivity, kterou jsme zaznamenávali, viz Obr. 15. Je zde uloženo, o jakou aktivitu se jednalo, v případě této práce jsme zaznamenali jízdu na kole. Je zde ještě ID tag, který slouží k identifikaci záznamu.

```
<Activities>
  <Activity Sport="Biking">
    <Id>2018-07-25T14:30:54.000Z</Id>
    <Lap StartTime="2018-07-25T14:30:54.000Z">
```

Obr. 15 TCX data část 2. (Zdroj: Autor)

Aktivity můžeme rozdělovat i na jednotlivé okruhy. V následujících několika řádcích souboru je shrnutí dat, které se týká právě jednoho okruhu, viz Obr. 16. V tomto shrnutí

je kromě začátku okruhu také, jak dlouho okruh trval, jeho vzdálenost v metrech, jaké nejvyšší rychlosti jsme dosáhli, jaké jsme měli tempo a kolik kalorií jsme spálili apod.

```
<Lap StartTime="2018-07-25T14:30:54.000Z">
  <TotalTimeSeconds>3009.0</TotalTimeSeconds>
  <DistanceMeters>19025.2</DistanceMeters>
  <MaximumSpeed>14.687000274658203</MaximumSpeed>
  <Calories>703</Calories>
  <Intensity>Active</Intensity>
  <Cadence>122</Cadence>
  <TriggerMethod>Manual</TriggerMethod>
```

Obr. 16 TCX data část 3. (Zdroj: Autor)

Nyní už se dostáváme k tagu <Track>, ve kterém se každou vteřinu zaznamenává tag <Trackpoint>. Do Trackpointu se zapisuje čas zaznamenání, současná pozice pomocí GNSS souřadnic, současná kadence, ujetá vzdálenost v metrech. Díky rozšíření ještě měříme rychlost a výkon. Příklad jednoho Trackpointu ukazuje Obr. 17

```
<Trackpoint>
  <Time>2018-07-25T15:01:08.000Z</Time>
  <Position>
    <LatitudeDegrees>50.72139230556786</LatitudeDegrees>
    <LongitudeDegrees>15.182474730536342</LongitudeDegrees>
  </Position>
  <AltitudeMeters>510.79998779296875</AltitudeMeters>
  <DistanceMeters>11302.400390625</DistanceMeters>
  <Cadence>71</Cadence>
  <Extensions>
    <ns3:TPX>
      <ns3:Speed>4.927000045776367</ns3:Speed>
      <ns3:Watts>301</ns3:Watts>
    </ns3:TPX>
  </Extensions>
</Trackpoint>
```

Obr. 17 TCX data část 4. (Zdroj: Autor)

Pokud uživatel, stejně jako v této práci, využil rozšíření bude na konci okruhu ještě celkové shrnutí hodnot naměřených pomocí tohoto rozšíření viz Obr. 18. V našem případě jsme měřili rychlost a výkon, proto zde bude průměrná rychlost a průměrný a maximální výkon. Maximální rychlost je již zmíněná výše v celkovém shrnutí okruhu.

```
<Extensions>
  <ns3:LX>
    <ns3:AvgSpeed>6.468999862670898</ns3:AvgSpeed>
    <ns3:MaxBikeCadence>61</ns3:MaxBikeCadence>
    <ns3:Steps>2978</ns3:Steps>
    <ns3:AvgWatts>239</ns3:AvgWatts>
    <ns3:MaxWatts>1214</ns3:MaxWatts>
  </ns3:LX>
</Extensions>
```

Obr. 18 TCX data část 5. (Zdroj: Autor)

Na závěr souboru je ještě několik tagů ohledně verze zařízení a podobně.

3.2.2 CSV

CSV (Comma separated values) je velmi populární formát využívaný pro výměnu a konvertování dat v různých tabulkových editorech. (Shafranovich, 2005) Z názvu vyplývá, že hodnoty jsou oddělovány separátorem, přesněji čárkou, to ale neplatí vždy, protože CSV má spoustu variací. Například v evropské verzi se jako separátor využívá středník. Proto ve výstupu této práce je formát CSV využit se středníky.

Tabulkový editor si tedy hodnoty uloží do jednotlivých buněk na řádku z řádku, který právě čte ze souboru. To následovně udělá i se zbytkem řádků v souboru. Pro lepší představu uvedeme příklad. Obr. 19 je napsán v textovém editoru a na Obr. 20 je interpretován tabulkovým editorem.

```
Time;Distance Meters;Cadence;Latitude Degrees;Longitude Degrees;Altitude Meters;Speed;Watts
25.07.2018 14:30:54;0;0;No data;No data;0;no data;0
25.07.2018 14:30:55;0;0;No data;No data;0;no data;0
25.07.2018 14:30:56;0;0;No data;No data;0;no data;0
25.07.2018 14:32:06;0;0;50,7740972097963;15,0763030070812;416,399993896484;0.0;no data
25.07.2018 14:32:07;0,0299999993294477;0;50,7740972936153;15,0763034261763;416,399993896484;0.0;0
25.07.2018 14:32:08;0,0900000035762787;0;50,7740972097963;15,0763042643666;416,399993896484;0.0;0
25.07.2018 14:32:09;0,119999997317791;0;50,7740967068821;15,0763046834618;416,399993896484;0.0;0
25.07.2018 14:32:10;0,119999997317791;0;50,7740940246731;15,0763034261763;416,399993896484;0.0;0
25.07.2018 14:32:11;1,07000005245209;0;50,7740862295032;15,0762980617583;416,399993896484;1.371999979019165;0
25.07.2018 14:32:12;2,76999998092651;0;50,7740770094097;15,0762784481049;416,399993896484;2.062000036239624;0
25.07.2018 14:32:13;4,98999977111816;0;50,7740692980587;15,0762493629009;416,399993896484;2.2769999504089355;0
25.07.2018 14:32:14;7,40999984741211;0;50,7740622572601;15,0762170087546;416,600006103516;2.3980000019073486;0
25.07.2018 14:32:15;10,4700002670288;0;50,7740570604801;15,0761742610484;416,399993896484;2.818000078201294;0
25.07.2018 14:32:16;13,9300003051758;0;50,7740518637002;15,0761259812862;416,600006103516;3.3399999141693115;0
25.07.2018 14:32:17;17,8099994659424;0;50,7740422245115;15,0760730914772;416,399993896484;3.6579999923706055;0
25.07.2018 14:32:18;21,9500007629395;0;50,7740291487426;15,0760181061924;416,399993896484;3.7320001125335693;0
25.07.2018 14:32:19;26,4200000762939;0;50,7740131393075;15,0759599357843;416,600006103516;4.031000137329102;0
25.07.2018 14:32:20;32,2200012207031;0;50,7739793602377;15,0758971553296;415,200012207031;5.383999824523926;0
25.07.2018 14:32:21;38,0499992370605;0;50,7739438209683;15,0758361350745;415;5.691999912261963;0
```

Obr. 19 Interpretace CSV dat textovým editorem (Zdroj: Autor)

Time	Distance Meters	Cadence	Latitude Degrees	Longitude Degrees	Altitude Meters	Speed	Watts
25.07.2018 14:30:54	0	0	No data	No data	0	no data	0
25.07.2018 14:30:55	0	0	No data	No data	0	no data	0
25.07.2018 14:30:56	0	0	No data	No data	0	no data	0
25.07.2018 14:32:06	0	0	50,7740972097963	15,0763030070812	416,399993896484	0.0	no data
25.07.2018 14:32:07	0,029999999329448	0	50,7740972936153	15,0763034261763	416,399993896484	0.0	0
25.07.2018 14:32:08	0,090000003576279	0	50,7740972097963	15,0763042643666	416,399993896484	0.0	0
25.07.2018 14:32:09	0,119999997317791	0	50,7740967068821	15,0763046834618	416,399993896484	0.0	0
25.07.2018 14:32:10	0,119999997317791	0	50,7740940246731	15,0763034261763	416,399993896484	0.0	0
25.07.2018 14:32:11	1,07000005245209	0	50,7740862295032	15,0762980617583	416,399993896484	1.371999979019165	0
25.07.2018 14:32:12	2,76999998092651	0	50,7740770094097	15,0762784481049	416,399993896484	2.062000036239624	0
25.07.2018 14:32:13	4,98999977111816	0	50,7740692980587	15,0762493629009	416,399993896484	2.2769999504089355	0
25.07.2018 14:32:14	7,40999984741211	0	50,7740622572601	15,0762170087546	416,600006103516	2.3980000019073486	0
25.07.2018 14:32:15	10,4700002670288	0	50,7740570604801	15,0761742610484	416,399993896484	2.818000078201294	0
25.07.2018 14:32:16	13,9300003051758	0	50,7740518637002	15,0761259812862	416,600006103516	3.3399999141693115	0
25.07.2018 14:32:17	17,8099994659424	0	50,7740422245115	15,0760730914772	416,399993896484	3.6579999923706055	0
25.07.2018 14:32:18	21,9500007629395	0	50,7740291487426	15,0760181061924	416,399993896484	3.7320001125335693	0
25.07.2018 14:32:19	26,4200000762939	0	50,7740131393075	15,0759599357843	416,600006103516	4.031000137329102	0
25.07.2018 14:32:20	32,2200012207031	0	50,7739793602377	15,0758971553296	415,200012207031	5.383999824523926	0
25.07.2018 14:32:21	38,0499992370605	0	50,7739438209683	15,0758361350745	415	5.691999912261963	0

Obr. 20 Interpretace CSV dat v tabulkovém editoru (Zdroj: Autor)

3.2.3 XML data z mobilní spirometrie

Data ve formátu XML se skládají z elementů, které mohou a nemusí mít atributy. Data z mobilní spirometrie, která byla použita v této práci jsou vytvořena pomocí schématu: schemas-microsoft-com:office:spreadsheet. Toto schéma využívají tabulkové editory od společnosti Microsoft. V následujících odstavcích budou vysvětleny nejdůležitější elementy dat z mobilní spirometrie.

<Styles> je element, pod kterým se skrývá mnoho dalších elementů. Všechny popisují finální vzhled při otevření v tabulkovém editoru. Tento element není pro cíl této práce důležitý, protože snaha této práce je převést XML do CSV a jelikož formát CSV nemá

prostředky pro úpravu vzhledu v tabulkovém editoru, není tedy pro tento element v CSV využití.

Další element je <Worksheet>, ve kterém se ukrývají jednotlivé tabulky v elementech <Table>. Každá tabulka je rozdělena na řádky (<Row>) a jednotlivé buňky (<Cell>). V jednotlivých buňkách jsou informace o datech v atributech elementu <Data>, ve kterém jsou hodnoty buněk. Příklad jednoho řádku je vyobrazen na Obr. 21

```
<Row>
  <Cell ss:StyleID="MeasurementDataTableTime"><Data ss:Type="String">0:06:00,420</Data></Cell>
  <Cell ss:StyleID="MeasurementDataTablePhases"><Data ss:Type="String">Zátěž</Data></Cell>
  <Cell ss:StyleID="MeasurementDataTableMarkers"><Data ss:Type="String"></Data></Cell>
  <Cell ss:StyleID="MeasurementDataTableValues3108"><Data ss:Type="Number">2.93905818242317</Data></Cell>
  <Cell ss:StyleID="MeasurementDataTableValues3210"><Data ss:Type="Number">31.9462845915562</Data></Cell>
  <Cell ss:StyleID="MeasurementDataTableValues3201"><Data ss:Type="Number">28.0801737174188</Data></Cell>
  <Cell ss:StyleID="MeasurementDataTableValues2018"><Data ss:Type="Number">104.666666666667</Data></Cell>
  <Cell ss:StyleID="MeasurementDataTableValues4001"><Data ss:Type="String">-</Data></Cell>
  <Cell ss:StyleID="MeasurementDataTableValues3208"><Data ss:Type="Number">21.9002794638408</Data></Cell>
  <Cell ss:StyleID="MeasurementDataTableValues3209"><Data ss:Type="Number">25.3071110118331</Data></Cell>
  <Cell ss:StyleID="MeasurementDataTableValues3110"><Data ss:Type="Number">0.865380463759799</Data></Cell>
  <Cell ss:StyleID="MeasurementDataTableValues3106"><Data ss:Type="Number">69.2236622222222</Data></Cell>
  <Cell ss:StyleID="MeasurementDataTableValues2001"><Data ss:Type="Number">2.42266666666667</Data></Cell>
  <Cell ss:StyleID="MeasurementDataTableValues2013"><Data ss:Type="Number">28.5733333333333</Data></Cell>
</Row>
```

Obr. 21 Úkázka XML dat z mobilní spirometrie (Zdroj: Autor)

3.3 Ošetření výjimek

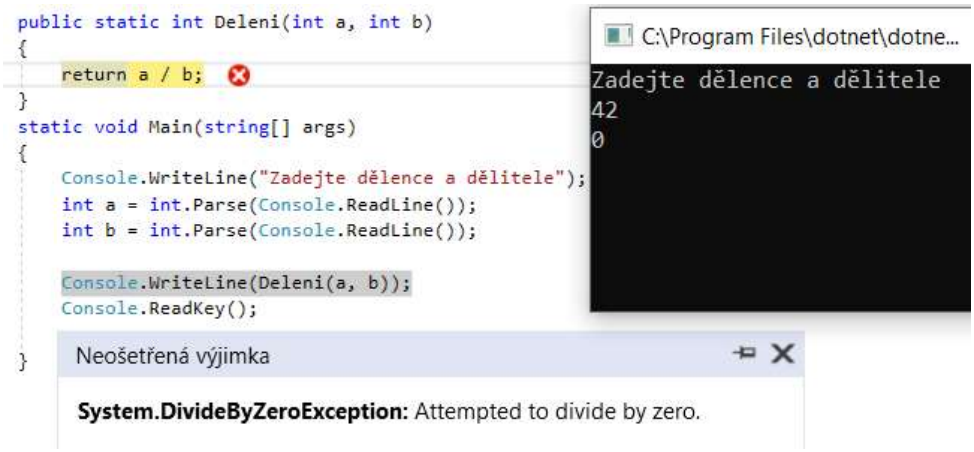
V následujících odstavcích budou popsány možnosti ošetření výjimek. Bude zde taky popsáno, jaké ošetření bylo v práci využito více, a proč.

Pokud v našem programu pracujeme se vstupy od uživatele, může se stát, že vstup od uživatele nebude dávat smysl nebo nebude vůbec existovat. Tyto chyby nejsou zapříčiněny funkčností programu, ale takto zranitelná místa je třeba ošetřit, aby program nepřestal pracovat. Pomocí ošetření výjimek jsme schopni nejen zabránit ukončení programu, kvůli chybě, ale také uživatele upozornit na situaci a nabídnout řešení. (Čápka, 2019)

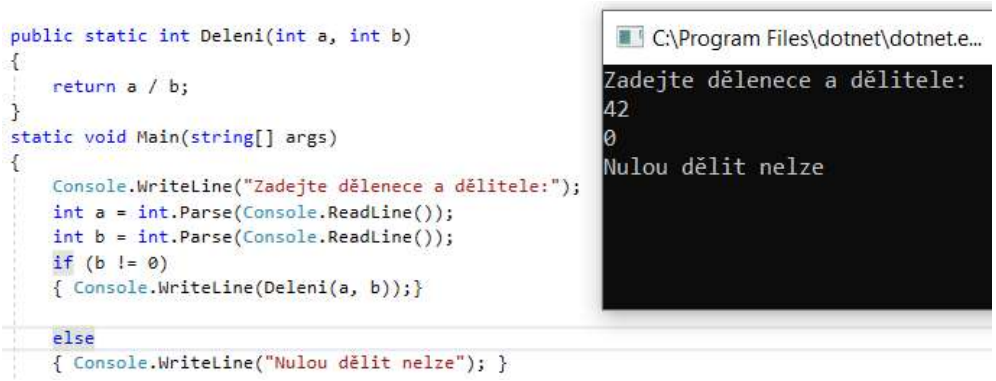
3.3.1 Aktivní ošetření výjimek

Budeme řešit dvě možnosti ošetření výjimek, aktivní a pasivní. Aktivní ošetření výjimek využívá podmínek. Kód, který by například nefungoval s určitým inputem, vložíme do bloku s podmínkou, která se postará o to, aby žádný špatný input do tohoto bloku vstoupit nemohl.

Nyní si uvedeme jednoduchý příklad pro lepší pochopení. Na Obr. 22 se uživatel pokouší dělit nulou. V tomto momentě by nastala chyba, protože nulou dělit nelze. Na Obr. 23 jsme výjimku ošetřili podmínkou, která zajišťuje, že dělitel nemůže být nula a pokud je upozorníme uživatele na fakt, že nulou dělit nelze.



Obr. 22 Výjimka před aktivním ošetřením (Zdroj: Autor)



Obr. 23 Výjimka po aktivním ošetřením (Zdroj: Autor)

3.3.2 Pasivní ošetření výjimek

Pasivní ošetření je dobré využívat u složitějších operací, u kterých je náročné pokrýt všechny výjimky, které by mohli nastat. K tomuto typu ošetření využíváme bloků try a catch, popřípadě ještě finnaly. Takto ošetřené výjimky jsou sice o něco pomalejší, ale jsou skvělé pro operace se soubory, protože mohou vyvolat velké množství výjimek. (Čápka, 2019)

Jelikož tato bakalářská práce je založena na práci se soubory, bylo pasivní ošetřování výjimek hojně využíváno.

Try blok

Do try bloku píšeme kód, u kterého máme podezření, že by mohl vyvolat výjimku. Tato část kódu je následně prováděna, dokud nenarazí na výjimku nebo je dokončena bez problémů. (Wagner, 2015)

Catch blok

Catch blok následuje po try bloku a může být jeden nebo více. Tento blok se provede, pouze pokud v bloku try narazil kód na výjimku, kterou má blok catch jako argument. Lze použít i bez argumentů pro zachycení jakékoliv výjimky. (Wagner, 2015)

Finnaly blok

Tento blok lze přidat k blokům try-catch. Tento blok se pustí vždy, ať už try skončí výjimkou nebo ne. Využívá se k zavírání souborů, uvolňování paměti, uložení nastavení a podobným akcím, které chceme vykonat, pokud kód v try skončí, jinak než jsme chtěli. (Čápka, 2019)

3.4 Parsing

C# je staticky typovaný jazyk, to znamená,že každá proměnná musí mít nejprve deklarovaný datový typ, až po té může být použita. Datové typy proměnných nelze později měnit, v případě porušení tohoto pravidla kompiler upozorní na chybu a program

nezkompiluje. Opakem staticky typovaného jazyka je dynamický typový jazyk, jeho zástupce je například PHP. (Čápka, 2019)

Parsování je proces, během kterého se převádí řetězec znaků do jiného formátu, se kterým se lépe pracuje. Například převod čísla bez desetiné čárky z textu do datového typu integer. Některé parsery jsou schopny porozumět textu pomocí syntaktické analýzy, tyto parsery jsou součástí kompileru. Díky tomu je kompiler schopen rozpoznat identifikátory, operátory aj.

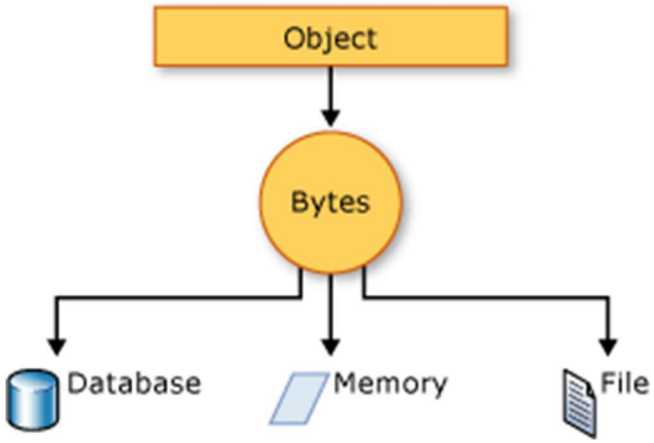
C# nabízí spoustu skvělých vlastních knihoven a metod pro parsování. Mezi tyto metody patří .parse a .tryparse. Tryparse při chybě vrátí hodnotu false, na rozdíl od parse , které skončí výjimkou. Obě výše zmíněné metody nemají problém s prázdnými znaky na začátku a na konci řetězce, pokud se ale prázdný znak nachází uprostřed řetězce, tak způsobí chybu. Syntaxe pro tyto metody je následující: datovýtyp.Parse(input);. Všechny znaky, kromě prázdných na začátku a na konci řetězce, musí být tvořeny příslušnými datovými číselnými typy (int, long, double, decimal atd.). (Wagner, 2019)

Mezi knihovny se řadí System.Convert. Třídy Convert jsou používány na převod do a ze základních datových typů. Mezi tyto datové typy patří Boolean, Char, Int, Single, Double, String, Datetime atd. (Wagner, 2019)

V této práci byly parsovány dva soubory. Jeden s příponou .tcx, který byl vytvořen zaznamenáním jízdy na kole během měření mobilní spirometrie. O parsování TCX dat se postarala knihovna OpenTcx(Yahch, 2019) a krátký námi připsaný kód. Druhý soubor ve formátu XML, ve kterém byla uložena přímo spirometrická data. U tohoto souboru bylo využito kódu, který data sparsoval do datasetu.

3.4.1 Serializace a deserializace XML

Serializace je proces, při kterém převádíme objekty do datového proudu bajtu. Využívá se k ukládání nebo odesílání do databáze, souboru či paměti. Pokud nám jde o zpětné vytvoření objektu, použijeme reverzní akci, která se nazývá deserializace. Schéma serializace a při otočením směru šipek desrializace vyobrazuje Obr. 24. (Wagner, 2020)



Obr. 24 Schéma serializace (Autor: Microsoft)

Výsledkem serializace je datový proud, a v tomto stavu se přenáší do databáze, souboru nebo paměti. Tento datový proud mimo jiné obsahuje i informace o typu objektu, názvu, jeho verzi a jazyku. (Wagner, 2020)

C# obsahuje třídy pro binární a XML serializaci. V této práci se zaměříme na XML serializaci. V XML serializaci se snažíme veřejné pole, vlastnosti objektu, návratové hodnoty a parametry uložit do datového proudu XML. Tento datový proud odpovídá konkrétnímu schématu XSD(XML Schema Definition Language). S využitím oboru názvů Systém.XML.Serialization máme k dispozici třídy jak pro XML serializaci, tak i deserializaci. (Wagner, 2020)

OpenTCX

OpenTCX je volně dostupná knihovna pro analyzování a generování TCX souborů v C#. Tuto knihovnu lze nalézt na adrese https://github.com/yahch/OpenTcx. V této práci byla tato knihovna využita k deserializaci TCX souborů, kromě části rozšíření (<extension>).

Jak funguje OpenTCX

Jak již bylo řečeno OpenTCX funguje jako deserializace TCX souborů. Pro nás jsou nejdůležitější dvě části této knihovny. Je to část TrainingCenterDatabase.cs a Tcx.cs. V Tcx.cs je část kódu zodpovědná za deserializaci a v TrainingCenterDatabase.cs jsou všechny objekty potřebné k deserializaci. OpenTcx bylo v práci využito k analýze TCX souboru, tedy deserializaci, pomocí metody AnalyzeTcxFile.

3.4.2 Parsování XML dat

V kapitole o datech XML z mobilní spirometrie bylo poukázáno na to, že data jsou rozdělena v elementech podle řádků a buněk. Tento XML formát je tedy již svým způsobem tabulka, proto byl pro tuto práci zvolen způsob parsování převodem XML dat do datasetu. Dataset je kolekce dat, která je rozdělená do jedné nebo více tabulek, má určitý počet sloupců a řádků, tím tvoří matici. K této matici poté přistupujeme pomocí indexů jednotlivých hodnot. Dataset nepožaduje mít ve všech sloupcích stejné proměnné, a to je velkou výhodou. Více o převodu XML dat do datasetu v kapitolách popisující program.

3.5 Kódování znaků

V objektově orientovaném programování existují abstraktní entity. Tyto entity mohou být následně různě interpretovány. Mezi tyto entity patří i znaky. Pokud chceme znak či řetězec znaků přeložit do sekvence bajtů, využijeme kodér, který tuto operaci provede. Naopak, když je naším záměrem dekódovat sekvenci bajtů do řetězce znaků použijeme dekodér. Právě kódování znaků určuje pravidla, jak má kodér a dekodér pracovat, a také přiřazuje k jednotlivým znakům jednotlivé kódy.

V .Net je několik tříd, které jsou určeny pro kódování znaků. Všechny tyto třídy dědí z abstraktní třídy System.Text.Encoding. V této třídě jsou nadefinované funkce, které jsou společné pro všechna kódovaní znaků, proto kdykoliv chceme nějakou z dědících tříd

použít, musíme přidat před kód: using Systém.Text.Encoding;. V následujících odstavcích si popíšeme pro nás nejdůležitější z těchto tříd.

ASCIIEncoding je třída pro ASCII kódování. K zakódování se používá pouze sedm bitů, proto je znaková sada velmi omezená a není dostačující pro mezinárodní potřeby.

UTF8Encoding je třida pro UTF-8 kódování. Každý znak je zakódován jako sekvence jednoho až čtyř bajtů. Má tedy mnohem větší sadu znaků nežli předchozí ASCII. V dnešní době je toto kódování velmi oblíbené a funguje dobře s mnoha operačními systémy. (Thraka, 2017)

Pokud si během psaní kódu nezvolíme žádnou třídu kódování, zvolí se automaticky znaková sada, která odpovídá nastavení systému.

V této práci bylo vzhledem k očekávanému jazyku uživatele využito kódování UTF8, které je vyhovující jak pro českého, tak i případného zahraničního uživatele.

3.6 Synchronizace

Tato práce se mimo jiné zaměřena také na synchronizaci fitness dat z cyklopočítače Garmin s daty z mobilní spirometrie. Následující kapitoly popisují, jak byla v této práci synchronizace provedena. Mimo jiné zde bude také popsán systém GNSS a jeho role v synchronizaci. Kromě samotné synchronizace se kapitoly zaměřují také na problematiku převzorkování.

3.6.1 Převzorkování

Protože soubor, který byl zaznamenán spirometrem má rozdílnou vzorkovací frekvenci než zařízení Garmin, kterým byly zaznamenány ostatní hodnoty, musíme nějakým způsobem tyto vzorkovací frekvence sjednotit. Využité GNSS zařízení měří hodnoty každou vteřinu a využitý spirometr data měří velmi proměnlivě v rozdílech dvou i tří vteřin. Zařízení Garmin má výrazně větší vzorkovací frekvenci, a proto jsme do programu implementovali dva možné způsoby, jak data ze spirometru převzorkovat aby vzorkovací

frekvence byla stejná jako u zařízení Garmin. Tyto dva způsoby jsou Sample and hold a lineární interpolace. Jaký způsob bude využit necháváme na uživateli.

V následujících dvou částech této práce budou popsány obě možnosti převzorkování. Převzorkování v této práci však silně souvisí se samotnou synchronizací, proto samotný algoritmus, včetně vývojového diagramu a části kódu, bude popsán až v kapitole Synchronizace cyklopočítače a mobilní spirometrie.

Sample and hold

Sample and hold funguje na velmi jednoduchém principu, který je popsán již v názvu. Vzorek, který naměříme používáme do té doby, než naměříme další. V této práci to znamená, že naše výsledná vzorkovací frekvence bude stejná jako v souboru TCX. Data z času Xs ze spirometrie neměníme dokud se v časech z TCX nedostaneme do času Xt, který se rovná Xs+1 . Výsledná křivka bude tedy vypadat, lidově řečeno, velmi zubatě.

Linearní interpolace

Při využití této možnosti se mezi data, která máme k dispozici, přidávají hodnoty vypočítané jednoduchou rovnicí. Rovnice nám zaručuje, že mezi nám známými hodnotami se přidávají postupně hodnoty, které se mění lineárně směrem k nám známé hodnotě, viz rovnice (2). Zjišťujeme hodnoty v intervalu mezi časem ze spirometrie X_s a $X_{(s+1)}$. X nám označuje současný čas z Garmin zařízení. $Y_{(s+1)}$ označuje hodnotu v čase $X_{(s+1)}$, stejně tak Y_s je hodnota v čase X_s. Y nám označuje výslednou hodnotu.

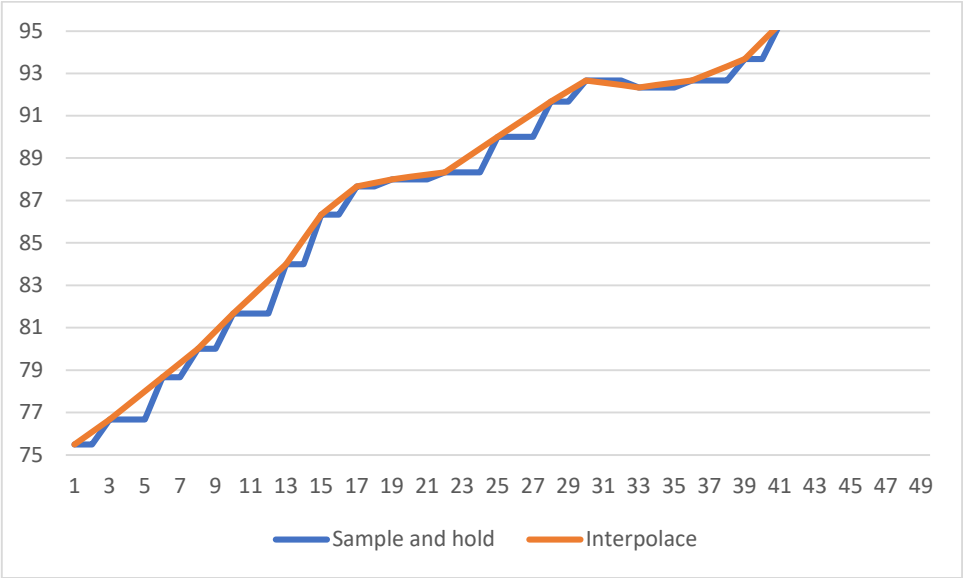
$$Y = Ys + (X - Xs) . \left(\left(\frac{Y(s+1) - Ys)}{(X(s+1) - Xs)}\right)\right) \qquad (2) \text{ (Hamerník, 2016, s.4)}$$

V programu k této bakalářské práci byla tato rovnice implementována v metodě interpolace vyobrazené na Obr. 25.

```
double interpolace(DateTime x0, DateTime x, DateTime x1, double y0, double y1)
{
    double y = y0 + (Convert.ToDouble((x - x0).Ticks) * ((y1 - y0) / Convert.ToDouble((x1 - x0).Ticks)));
    return y;

}
```

Obr. 25 Metoda interpolace (Zdroj: Autor)



Graf 1 Rozdíl mezi zvolenými převzorkováními

Pro názornost lze vidět na Graf 1 rozdíl mezi převzorkováním Sample and hold a lineární interpolací. Byl využit záznam tepové frekvence během měření mobilní spirometrie.

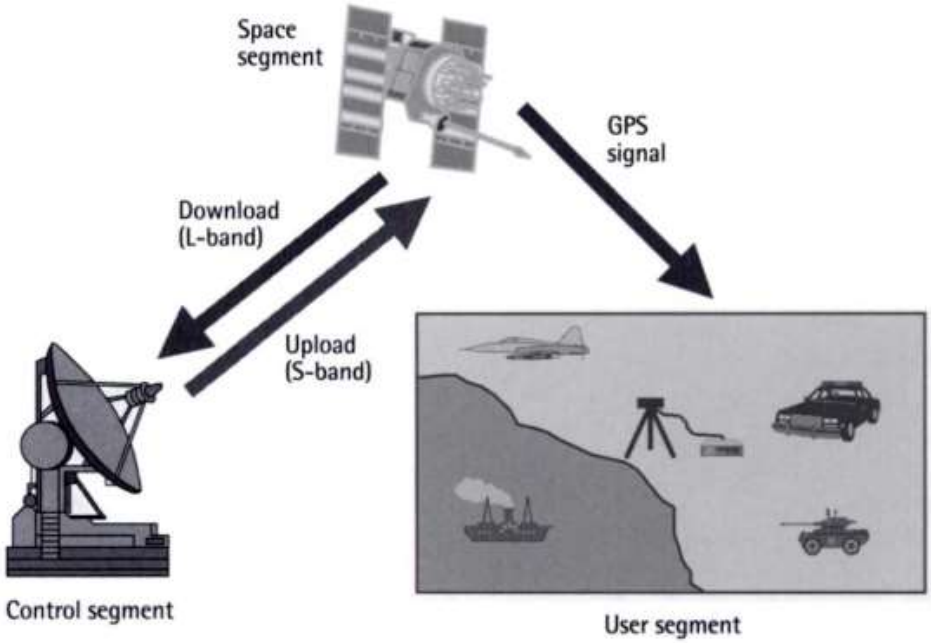
3.6.2 GNSS a čas

V následujících odstavcích je krátce vysvětleno, jak systém GNSS funguje a jak práce tohoto systému souvisí s časem. Někdy lidé mluví o GNSS jako o GPS, to je však pouze americkou podmnožinou GNSS.

Systém GNSS lze rozdělit do tří segmentů. První segment se nazývá vesmírný, a skládá se z 24 družic, které zajišťuji nepřetržité informace o své poloze a času. Rozestavění těchto družic je navrženo tak, aby přijímač, který může být kdekoliv na planetě, mohl přijímat, alespoň čtyři satelity. Čím více satelitů má přijímač k dispozici tím větší přesnost polohy. Druhý segment se nazývá ovládací, tento segment tvoří celosvětová síť

sledovacích stanic. Ovládací segment má za úkol hlídat pozici družic, korigovat atomové hodiny na satelitech atd., tyto informace jsou následně odesílány do družic. Poslední segment jsou samotní uživatelé. GNSS bylo původně vyvinuto pro armádu, později však bylo zpřístupněno i pro veřejnost. Nyní tedy stačí mít GNSS přijímač a člověk se může zcela zdarma připojit k tomuto systému. (El-Rabany, 2006)

Atomové hodiny v satelitech udržuje ze země ve správném čase ovládací segment. Na přesnosti hodin závisí i přesnost polohy. (El-Rabany, 2006) Pro lepší pochopení provázanosti segmentů, jsou na Obr. 26 vyobrazeny jednotlivé segmenty a komunikace mezi nimi.



Obr. 26 GNSS segmenty (Zdroj: Ahmed El-Rabbany, 2006, s.3)

Samotné zjištění polohy spočívá ve výpočtu vzdáleností od jednotlivých satelitů pomocí času, který satelity posílají. K nalezení správné polohy jsou třeba minimálně čtyři satelity. Protože satelity sdílí svůj přesný čas z atomových hodin, svůj identifikátor a polohu může náš přijímač vypočítat vzdálenost od jednotlivých satelitů. To vytváří okolo satelitů pomyslné koule a tam kde se koule protínají je naše poloha. Díky tomuto systému jsme schopni zjistit naše souřadnice. A protože satelity vysílají i čas, je možné tento čas synchronizovat i do našeho přijímače. Celý systém se ještě musí vypořádat s relativitou času a vlivem ionosféry, ale to pro tuto práci není důležité.

3.6.3 Synchronizace cyklopočítače a mobilní spirometrie

Cílem práce bylo i synchronizovat TCX fitness data s daty získanými z mobilní spirometrie, bohužel data z mobilní spirometrie neměla formát GNSS. Zařízení Garmin, které bylo využito, k naměření TCX fitness dat funguje i jako GNSS přijímač a má tedy i čas stanovený pomocí GNSS. Zařízení využité při snímání mobilní spirometrie sice disponuje GNSS modulem, ale jediný čas, který je ve finálních datech ve formátu GNSS, je startovací čas. Proto jsme před samostatnou synchronizací nejdříve převedli čas na stejný formát.

Přesný formát času z mobilní spirometrie je následující: h:mm:ss,ms. Po spuštění měření začíná čas od nuly a vzorkovací frekvence je silně proměnlivá. Naopak formát času z cyklopočítače Garmin je nasledující: yyy-MM-DDTHH:mm:ss, tedy jakmile začne měření, čas se začne zaznamenávat v tomto formátu a až na nějaké nesrovnalosti na začátku měření je vzorkovací frekvence 1Hz. Změny formátu času jsme dosáhli přičítáním času z mobilní spirometrie ke zvolené nule z cyklopočítače. Přičítaný čas byl zaokrouhlen na sekundy, aby byl formát naprosto stejný. Do programu byly implementovány dva způsoby nastavení začátku měření mobilní spirometrie ve formátu stejném jako ze souboru TCX. V prvním případě jsme využili velkých rozdílů ve vzorkovací frekvenci v časech TCX před začátkem měření, tedy jakmile se vzorkovací frekvence ustálí, program určí toto místo jako nulu. Pokud by program žádné takové místo nenašel, je za nulu považován první vzorek. Druhý způsob je input od uživatele. Uživateli poté, co vybere soubor TCX, který chce synchronizovat, budou ukázány všechny časy z TCX souboru, poté stačí z těchto časů vybrat nulu. Po přeformátování času přecházíme k algoritmu popsaném v další kapitole.

3.6.4 Využitý algoritmus synchronizace

V této práci byly využity dva algoritmy. V konečném běhu programu se využije pouze jeden, podle výběru uživatele. Algoritmy jsou rozdílné, kvůli možnostem převzorkování. Na začátku tedy vždy kontrolujeme, jaký typ převzorkování byl vybrán uživatelem. Poté už se přesouváme k samotné synchronizaci s převzorkováním. Základní princip

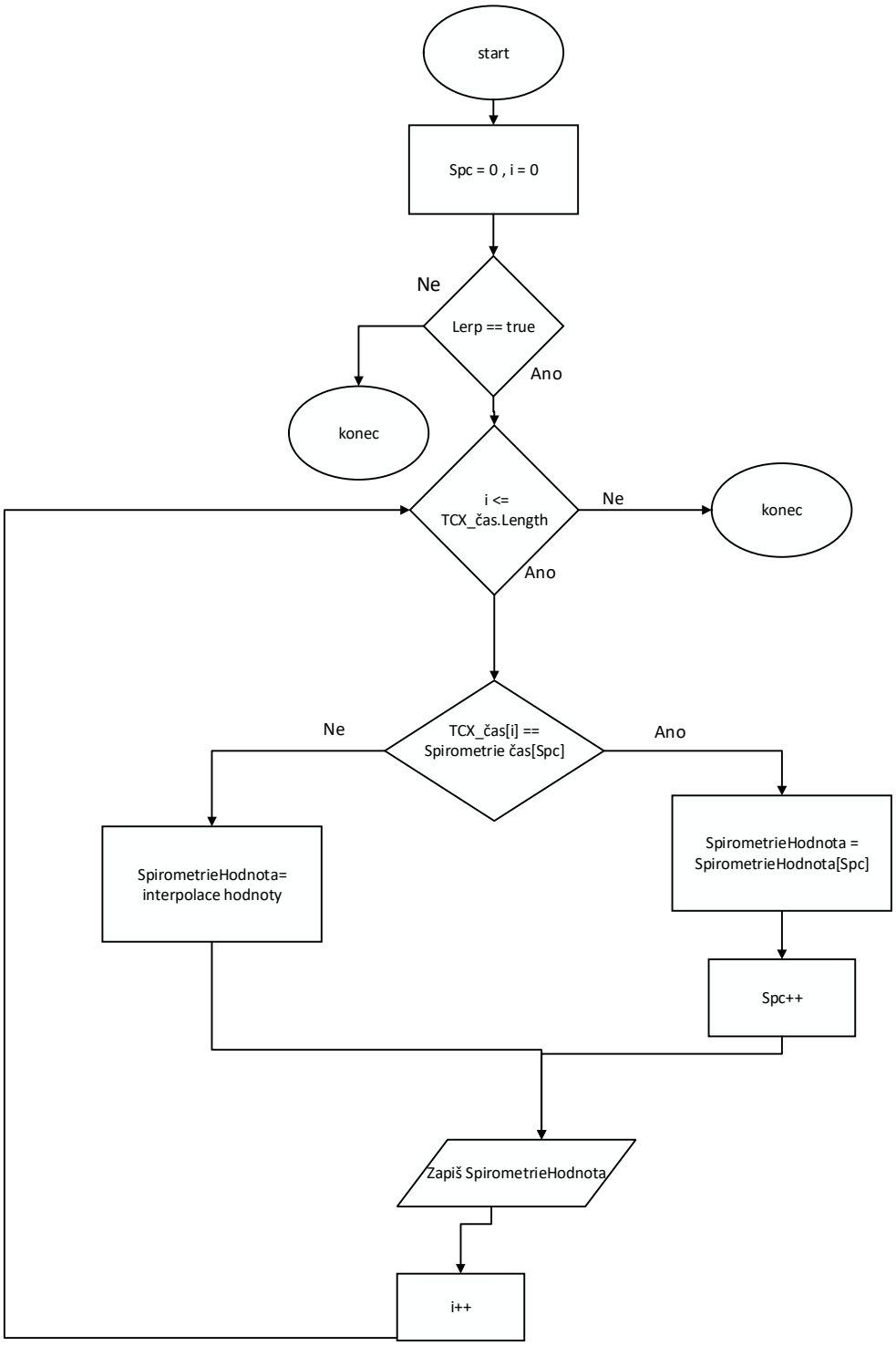
algoritmu je postupné porovnávání časů ze zařízení Garmin s časy ze zařízení mobilní spirometrie, které byly přeformátovány, aby měly stejný formát času jako zařízení Garmin. Pokud se časy rovnají, zapíše se hodnota s aktuálním indexem z listu hodnot a index se inkrementuje o jedna. Když se časy nerovnají, tak se hodnota vybere podle zvoleného převzorkování. Při zvolení lineární interpolace se jednotlivé hodnoty vypočítají pomocí vzorce, viz kapitola Lineární interpolace, dokud se časy opět nebudou rovnat. Při zvolení Sample and hold se bude stále zapisovat stejná hodnota, dokud se časy opět nebudou rovnat.

Obrázky níže ukazují výše popsané principy jak ve vývojovém diagramu (viz Obr. 27 a Obr. 28), tak v kódovém zpracování algoritmu (viz Obr. 29 a Obr. 30). Pro udržení přehlednosti, nebyla ve vývojovém diagramu zobrazena všechna ošetření výjimek, které se nachází v kódovém zpracování. Pro správné pochopení algoritmu však tyto diagramy stačí.

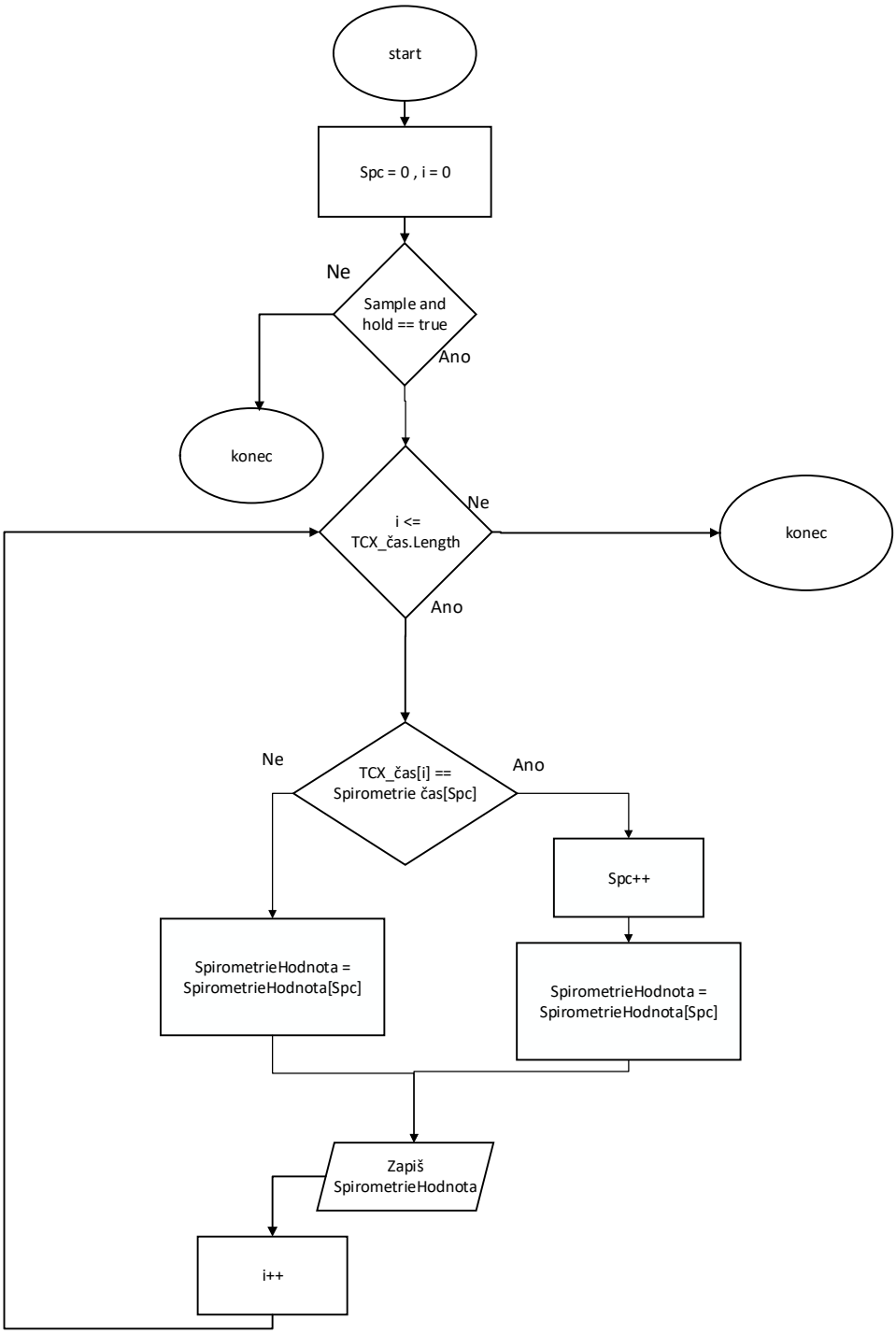
3.6.5 Další možnosti synchronizace

V této práci byl ke snímání spirometrických dat využit přístroj bez GPS modulu. Řešení této práce tedy počítá s daty z přístroje bez GPS modulu. Pokud uživatel použije GPS modul, měli by časy Garmin zařízení a spirometrického zařízení souhlasit. Pokud mají zařízení i stejnou vzorkovací frekvenci, může uživatel využít například výstup této práce k přeformátování TCX do CSV a následně data sám jednoduše synchronizovat nalezením souhlasných časů a zkopírováním správných hodnot v tabulkovém editoru.

Pokud formát času souhlasí, ale vzorkovací frekvence ne, pro co nejednoduší synchronizaci doporučujeme využít Excel. Krátce a stručně popíšeme možný postup. Začněte převedením TCX do CSV. Po převedení využijte funkci SVYHLEDAT, v anglické verzi VLOOKUP. Pokud chcete udělat přehlednou tabulku, lze využít podmínek tak, aby nenalezené hodnoty měly Vámi zvolenou úpravu.



Obr. 27 Vývojový diagram synchronizace za použití lineární interpolace k převzorkování (Zdroj: Autor)



Obr. 28 Vývojový diagram synchronizace za použití Sample and hold převzorkování (Zdroj: Autor)

```
if (Lerp == true)
{
    if (Spc < SpirometrieCas.Count)
    {
        if (data.Activities.Activity[0].Lap[0].Track[i].Time != SpirometrieCas[Spc] && startovano == true)
        {
            try
            {
                if (Spc >= 1)
                {
                    sw.Write(Faze[Spc] + ";");
                    sw.Write(Znacka[Spc] + ";");
                    sw.Write(interpolace(SpirometrieCas[Spc - 1], data.Activities.Activity[0].Lap[0].Track[i].Time, SpirometrieCas[Spc], VO2[Spc - 1] , VO2[Spc]) + ";");
                    sw.Write(interpolace(SpirometrieCas[Spc - 1], data.Activities.Activity[0].Lap[0].Track[i].Time, SpirometrieCas[Spc], VO2Kg[Spc - 1] , VO2Kg[Spc]) + ";");
                    sw.Write(interpolace(SpirometrieCas[Spc - 1], data.Activities.Activity[0].Lap[0].Track[i].Time, SpirometrieCas[Spc], VO2HR[Spc - 1] , VO2HR[Spc]) + ";");
                    sw.Write(interpolace(SpirometrieCas[Spc - 1], data.Activities.Activity[0].Lap[0].Track[i].Time, SpirometrieCas[Spc], Tf[Spc - 1] , Tf[Spc]) + ";");
                    sw.Write(WR[Spc] + ";");
                    sw.Write(interpolace(SpirometrieCas[Spc - 1], data.Activities.Activity[0].Lap[0].Track[i].Time, SpirometrieCas[Spc], VEO2[Spc - 1], VEO2[Spc]) + ";");
                    sw.Write(interpolace(SpirometrieCas[Spc - 1], data.Activities.Activity[0].Lap[0].Track[i].Time, SpirometrieCas[Spc], VECO2[Spc - 1] , VECO2[Spc]) + ";");
                    sw.Write(interpolace(SpirometrieCas[Spc - 1], data.Activities.Activity[0].Lap[0].Track[i].Time, SpirometrieCas[Spc], RER[Spc - 1]  , RER[Spc]) + ";");
                    sw.Write(interpolace(SpirometrieCas[Spc - 1], data.Activities.Activity[0].Lap[0].Track[i].Time, SpirometrieCas[Spc], VE[Spc - 1], VE[Spc]) + ";");
                    sw.Write(interpolace(SpirometrieCas[Spc - 1], data.Activities.Activity[0].Lap[0].Track[i].Time, SpirometrieCas[Spc], VT[Spc - 1], VT[Spc]) + ";");
                    sw.WriteLine(interpolace(SpirometrieCas[Spc - 1], data.Activities.Activity[0].Lap[0].Track[i].Time, SpirometrieCas[Spc], BF[Spc - 1], BF[Spc]) + ";");
                }
            }
            catch (Exception)
            {
                sw.WriteLine("Chyba interpolace");
            }
        }
    else if (data.Activities.Activity[0].Lap[0].Track[i].Time == SpirometrieCas[Spc])
    {
        if (startovano == false)
        {
            startovano = true;
        }
        try
        {
            sw.Write(Faze[Spc] + ";");
            sw.Write(Znacka[Spc] + ";");
            sw.Write(VO2[Spc] + ";");
            sw.Write(VO2Kg[Spc] + ";");
            sw.Write(VO2HR[Spc] + ";");
            sw.Write(Tf[Spc] + ";");
            sw.Write(WR[Spc] + ";");
            sw.Write(VEO2[Spc] + ";");
            sw.Write(VECO2[Spc] + ";");
            sw.Write(RER[Spc] + ";");
            sw.Write(VE[Spc] + ";");
            sw.Write(VT[Spc] + ";");
            sw.WriteLine(BF[Spc] + ";");
            Spc++;

        }
        catch (Exception)
        {
            sw.WriteLine("Žádná další data ze spirometrie");
        }
    }
    else
    {
        sw.WriteLine("Spirometrie nezačala;");
    }
  }

else
{
    try
    {
        sw.WriteLine(0);

    }
    catch (Exception)
    {
        sw.WriteLine("Chyba;");

    }

}
```

Obr. 29 Kódové zpracování synchronizace za použití linearní interpolace k převzorkování (Zdroj: Autor)

```
if (SaH == true)
{
    if (data.Activities.Activity[0].Lap[0].Track[i].Time>=SpirometrieCas[1])
    {
        if (Spc < SpirometrieCas.Count-1)
        {
            if (data.Activities.Activity[0].Lap[0].Track[i].Time == SpirometrieCas[Spc+1] && konecSpiro==false)
            {
                try
                {   Spc++;
                    sw.Write(Faze[Spc] + ";");
                    sw.Write(Znacka[Spc] + ";");
                    sw.Write(VO2[Spc] + ";");
                    sw.Write(VO2Kg[Spc] + ";");
                    sw.Write(VO2HR[Spc] + ";");
                    sw.Write(Tf[Spc] + ";");
                    sw.Write(WR[Spc] + ";");
                    sw.Write(VEO2[Spc] + ";");
                    sw.Write(VECO2[Spc] + ";");
                    sw.Write(RER[Spc] + ";");
                    sw.Write(VE[Spc] + ";");
                    sw.Write(VT[Spc] + ";");
                    sw.WriteLine(BF[Spc] + ";");

                   }

                catch (Exception)
                {
                    sw.WriteLine("žádná další data");
                }
              if (Spc+1==SpirometrieCas.Count-1)
            {
                konecSpiro = true;
            }
            }
            else if(data.Activities.Activity[0].Lap[0].Track[i].Time == SpirometrieCas[SpirometrieCas.Count-1] && konecSpiro == true)
            {
                Spc++;
                sw.Write(Faze[Spc] + ";");
                sw.Write(Znacka[Spc] + ";");
                sw.Write(VO2[Spc] + ";");
                sw.Write(VO2Kg[Spc] + ";");
                sw.Write(VO2HR[Spc] + ";");
                sw.Write(Tf[Spc] + ";");
                sw.Write(WR[Spc] + ";");
                sw.Write(VEO2[Spc] + ";");
                sw.Write(VECO2[Spc] + ";");
                sw.Write(RER[Spc] + ";");
                sw.Write(VE[Spc] + ";");
                sw.Write(VT[Spc] + ";");
                sw.WriteLine(BF[Spc] + ";");
            }
            else
            {
                try
                {

                 sw.Write(Faze[Spc] + ";");
                 sw.Write(Znacka[Spc] + ";");
                 sw.Write(VO2[Spc] + ";");
                 sw.Write(VO2Kg[Spc] + ";");
                 sw.Write(VO2HR[Spc] + ";");
                 sw.Write(Tf[Spc] + ";");
                sw.Write(WR[Spc] + ";");
                sw.Write(VEO2[Spc] + ";");
                sw.Write(VECO2[Spc] + ";");
                sw.Write(RER[Spc] + ";");
                sw.Write(VE[Spc] + ";");
                sw.Write(VT[Spc] + ";");
                sw.WriteLine(BF[Spc] + ";");

                }
                catch (Exception)
                {
                    sw.WriteLine("žádná další data ze spirometrie" + ";");

                }
            }
        }
        else
        {
            sw.WriteLine(0);

        }
    }
        else
        {
            sw.WriteLine("Zatím žádná data ze spirometrie;");
        }
    }
```

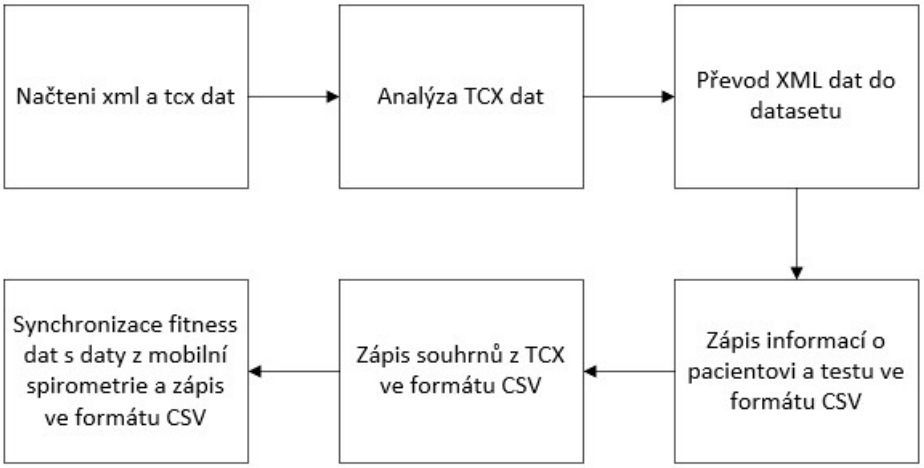
Obr. 30 Kódové zpracování synchronizace za použití Sample and hold převzorkování (Zdroj: Autor)

3.7 Popis programu

V této kapitole budou v jednotlivých podkapitolách uvedeny některé zásadní podrobnosti pro funkčnost programu, který je výstupem práce. Vytvořený program má dva možné výstupy. První výstup je čisté převedení TCX souboru do CSV. Druhý možný výstup je převedení souborů TCX a XML do CSV včetně synchronizace dat.

3.7.1 Blokové schéma programu

Bloková schémata programu podle výběru uživatele (viz Obr. 31 a Obr. 32). Jednotlivé bloky jsou rozebrány v následujících kapitolách.



Obr. 31 Blokové schéma převodu TCX a XML do CSV včetně synchronizace (Zdroj: Autor)

Obr. 32 Blokové schéma převodu TCX do CSV (Zdroj: Autor)

Načtení Xml a Tcx

Aby se s daty dalo pracovat musíme je nejdříve načíst. K načtení dat v této práci bylo využito třídy Systém.IO.

Analýza TCX dat

Tento blok se zabývá parsováním TCX dat. K tomuto účelu byla využita knihovna OpenTCX, která je popsána v kapitole Parsing v podkapitole OpenTCX. Tento blok slouží k deserializaci TCX dat, po této analýza máme přístup roztříděným datům s přiřazenými datovými typy.

Parsing <Extension>

V předchozích kapitolách bylo zmíněno, že k parsování souboru TCX byla využita knihovna OpenTcx, to je pravda, ale tato knihovna není schopna sparsovat i celý tag <Extension>. Proto jsme vytvořili krátký kód, který se nám o zpracování tohoto tagu postará.

V kódu vytvořeném pro vyhledání všech hodnot z tagu <Extension> jsme využili metodu getBetween (Oscar,2012), která vyhledá string mezi jedním a druhým námi zvoleným

stringem. Tato metoda byla pro námi vytvořený algoritmus zásadní a je vyobrazena na Obr. 33. Druhá důležitá metoda pro využitý algoritmus byla metoda Split. Split rozdělí námi zvolený řetězec do dvou podřetězců. Stačí zadat oddělovač, který je následně vyhledán ve zvoleném řetězci, vše před oddělovačem je první podřetězec, vše za oddělovačem je druhý podřetězec.

```
public static string getBetween(string strSource, string strStart, string strEnd)
{
    int Start, End;
    if (strSource.Contains(strStart) && strSource.Contains(strEnd))
    {
        Start = strSource.IndexOf(strStart, 0) + strStart.Length;
        End = strSource.IndexOf(strEnd, Start);
        return strSource.Substring(Start, End - Start);
    }
    else
    {
        return "no data";
    }
}
```

Obr. 33 Getbetween metoda (Zdroj: Oscar)

Po porozumění dvěma výše zmíněným metodám, není těžké vytvořit využitý algoritmus. Celý TCX dokument jsme si rozdělili pomocí metody split, aby v každém podřetězci byl pouze jeden tag <Extension>. Poté jsme postupně prohledali všechny vytvořené podřetězce a využili metodu getBetween pro nalezení jednotlivých hodnot podtagů jako jsou <Speed> a <Watt>. Vytvořený kód viz Obr 34.

```
string text = sr.ReadToEnd();
string[] Ns3Tpx = new string[] { "</Extensions>" };
string[] extension = text.Split(Ns3Tpx, StringSplitOptions.None);
int lengths = extension.Length;
string Speed;
string Watts;
string[] Wattss = new string[lengths];

string[] Speeds = new string[lengths];
for (int s = 0; s < length; s++)
{
    Speed = Replace(getBetween(extension[s], "<ns3:Speed>", "</ns3:Speed>"));
    Watts = Replace(getBetween(extension[s], "<ns3:Watts>", "</ns3:Watts>"));

    if (Speed == "no data")
    {
        Speeds[s] = "žádná data";
    }
    else
    {
        Speeds[s] = Speed;
    }
    if (Watts == "no data")
    {
        Wattss[s] = "žádná data";
    }
    else
    {
        Wattss[s] = Watts;
    }
}

string AvgSpeed = Replace(getBetween(text, "<ns3:AvgSpeed>", "</ns3:AvgSpeed>"));
string AvgWatts = Replace(getBetween(text, "<ns3:AvgWatts>", "</ns3:AvgWatts>"));
string MaxWatts = Replace(getBetween(text, "<ns3:MaxWatts>", "</ns3:MaxWatts>"));
string MaxBikeCadence = Replace(getBetween(text, "<ns3:MaxBikeCadence>", "</ns3:MaxBikeCadence>"));
string Steps = Replace(getBetween(text, "<ns3:Steps>", "</ns3:Steps>"));
```

Obr 34 Zpracování tagu <Extension> (Zdroj: Autor)

Převod XML dat do datasetu

Jak funguje tento typ parsování a proč byl využit zrovna v této práci popisuje kapitola Parosvání XML dat. Pro tento účel byla zvolena vytvořená třída ze zdroje (ColinBashBash, 2012). Licence této třidy dovoluje tento kód bez problému využít pro tuto práci. Tato třída byla vytvořena i pro XLS import do datasetu, tato část kódu však není pro účely této práce využitelná, a proto byla smazána. Jak třídu a požadovanou metodu použít, včetně příkladu přístupu k jednotlivým hodnotám viz Obr. 35.

```
// Spirometrie xml čtení  a převod do Datasetu
FileStream File = new FileStream(@xml, FileMode.Open, FileAccess.Read,
FileShare.Read);
DataSet excel = ExcelImport.ImportExcelXML(File, true, true);
string variable = excel.Tables[0].Rows[120][6].ToString(); // příklad přístup k jednotlivým hodnotám v datasetu
//
```

Obr. 35 Zpracování hodnot ze spirometrie do datasetu (Zdroj: Autor)

Zápis

K zápisu byla využita třída streamwriter. V této třídě bylo nastaveno kódování znaků na UTF8. Proč zrovna toto kódování, a jak kódování znaků funguje je popsáno v kapitole Kódování znaků. Abychom dosáhli v konečném výsledku formátu CSV, stačí mezi každou hodnotu zapsat středník na řádku. Řádky ukončujeme příkazem WriteLine(„ ; “);.

Zápis XML informací o pacientovi

V tomto bloku bylo využito cyklu while. Jelikož víme, jaký nadpis mají hodnoty zaznamenané mobilní spirometrií, stačí zapisovat všechny hodnoty, dokud nenarazíme na jasně definovaný nadpis. Použit byl ještě druhý while cyklus, který zaručuje, že program nebude zapisovat prázdná pole po konci řádku, který má prázdná pole, protože je zapsán maticí a ta musí mít stejný počet sloupců na každém řádku. Vyobrazení celého while cyklu zaměřeného na hledání začátku hodnot a zapsání dat o pacientovi viz Obr. 36.

```
while (excel.Tables[0].Rows[zacatek][p].ToString() != "t")
{
    while (ColumnsEnd == false)
    {
        sw.Write(excel.Tables[0].Rows[zacatek][p] + ";");

        p++;
        if (p < excel.Tables[0].Columns.Count)
        {


        if (excel.Tables[0].Rows[zacatek][p].ToString() == "" && excel.Tables[0].Rows[zacatek][p+1].ToString() == "")
            {
            ColumnsEnd = true;
            }

    }

        else if (p >= excel.Tables[0].Columns.Count || ColumnsEnd == true )
        {

            ColumnsEnd = true;

        }
    }
    ColumnsEnd = false;
    sw.WriteLine(";");
    zacatek++;
    p = 0;
}
```

Obr. 36 Zápis informací o pacientovi s hledáním začátku hodnot (Zdroj:Autor)

Zápis hodnot

Nejdříve zapíšeme souhrny hodnot z TCX souboru, které se skládají z průměrů a maxim dat, z ujeté vzdálenosti, celkového času apod. K zápisu všech hodnot byl využit for cyklus, který bere všechny hodnoty z TCX souboru a podle volby uživatele je pouze zapíše nebo synchronizuje se zvoleným převzorkováním.

3.7.2 Využité metody

Replace

Tato jednoduchá metoda najde char ve stringu a zamění ho za námi zvolený char. Tuto metodu jsme využili hlavně pro změnu desetinné čárky, která je v anglickém jazyce vytvořena tečkou. Pokud bychom v číselných datech tuto změnu neprovedli, musel by koncový uživatel využívající tabulkový editor s nastavenou češtinou tuto změnu provést sám, jinak by nemohl s daty správně pracovat. Metoda nebyla využita pouze k ulehčení

práce koncovému uživateli, ale hrála také roli během parsování. Metoda Replace viz Obr. 37

```
public static string Replace(string Source)
{
    string a = Source;

    a = a.Replace(".", ",");

    return a;
}
```

Obr. 37 Metoda Replace (Zdroj: Autor)

Formátování času

Protože během synchronizace porovnáváme dva časy, je třeba aby tyto časy byly ve stejném formátu. Tato podmínka není v původních datech splněna, proto byla do programu implementovaná metoda, která přeformátuje čas z mobilní spirometrie na stejný formát jako jsou data z GNSS zařízení Garmin.

Konečný požadovaný formát času je yyy-MM-DDTHH:mm:ss na rozdíl od HH:mm:ss,ms. Princip metody je jednoduchý, stačí vyhledat nulu, která je v souboru spirometrie označena, jako „Počáteční čas“. Tento počáteční čas není v požadovaném formátu. Jakmile nalezneme správný stačí k tomuto počáteční mu času přičíst uplynulý časový interval z mobilní spirometrie. Nakonec musíme pouze zaokrouhlit milisekundy na celé sekundy a máme stejný formát času. Takové zaokrouhlení nám neovlivní hodnoty natolik, aby to pro výslednou přesnost představovalo problém. Metodu vytvořenou pro tuto práci můžete vidět na Obr. 38

```
public static DateTime FormatCas (string cas,DateTime start) {
    TimeSpan ts = TimeSpan.Parse(cas);
    var rounded = TimeSpan.FromSeconds(Math.Round(ts.TotalSeconds, 0)); //zaokrouhlení ms
    DateTime Spirocas = start.Add(rounded);

    return Spirocas;
}
```

Obr. 38 Formátování času (Zdroj: Autor)

3.7.3 Grafické uživatelské rozhraní

GUI je rozhraní vytvořené pro přístup uživatele k programu. Pro GUI v této práci bylo využito oboru názvů Windows.Forms. Protože program má jen dva možné outputy byla zde snaha vytvořit co nejednoduší uživatelské rozhraní, které uživatel snadno pochopí a bude s ním lehce pracovat.

BackgroundWorker

Třída BackgroundWorker se využívá, pokud je požadováno, aby aplikace dělala více věcí zároveň. Jelikož BackgroundWorker je v podstatě multithreading, Je více možností jak multithreadingu docílit, pro účely této práce však BackgroundWorker stačí. BackgroundWorker spustí tedy úkon backgroundworkeru na samostatném vlákně. Tohoto faktu bylo v aplikaci využito, aby se během zpracování požadovaného úkonu aplikace nezastavila. Abychom upozornili uživatele, že úkon je dokončen, byl přidán messegebox, který se objeví, jakmile backgroundworker dokončí námi zadaný úkon. Stejně tak bude uživatel upozorněn, pokud se zadaný úkon nepodaří splnit. Také jsme využili spojení BackgroundWorkeru s progressbarem, který uživatele upozorní na probíhající úkon.

OpenFileDialog a SaveFiledialog

Oba dva v názvu zmíněné dialogy slouží k usnadnění hledání cesty k souboru, ať už k existujícímu v případě openfiledialogu nebo k souboru, který teprve tvoříme v případě savefiledialogu. Pokud by uživatel chtěl cestu k souboru pouze zkopírovat a nevyužít

těchto dialogů, může zapsat cestu do příslušně označených textboxů. Touto cestou však nelze využít vybrání počátečního času z TCX a čas bude vybrán automaticky.

3.7.4 Omezení vytvořeného programu

Program byl vytvořen tak, aby odpovídal zadání práce, proto není naprosto univerzální a je třeba počítat s určitými omezeními. Omezení, budou popsány v dalších odstavcích.

Následující omezení je třeba dodržet pouze pokud chce uživatel využít vytvořené funkce synchronizace. Čas v souboru z mobilní spiromterie musí být ve stejném formátu, jako je popsaný v kapitole Synchronizace cyklopočítače a mobilní spirometrie. V zařízení Garmin musí zvolit vzorkovací frekvenci tak aby odpovídala 1Hz, minimálně po startu žádoucího měření. Takže by neměl využívat úsporných funkcí zařízení. Zařízení Garmin musí být po celou dobu měření mobilními spirometry zapnuto.

U pouhého převodu TCX do CSV, nebyly zaznamenány problémy. Program byl koncipován modulárně a nepokrývá zcela všechny možné tagy, byly využity jen tagy pro účely této práce. Další tagy lze do programu, v případě zájmu, lehce doplnit.

3.8 Návod k vytvořenému programu

Ovládací prvky jsou označeny na Obr. 39 a nad obrázkem popsána jejich funkce.

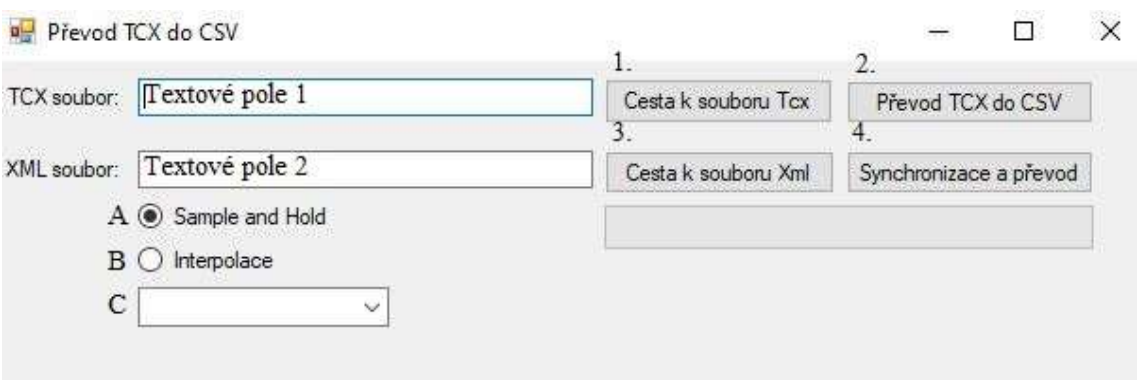
Textové pole 1 – Do tohoto textového pole zapište cestu k TCX souboru, který chcete přeformátovat

Textové pole 2 – Pokud chcete přeformátování dat včetně synchronizace s daty z mobilní spirometrie

Tlačítko 1. a 3. – Pokud nechcete zapisovat ručně cestu k souboru využijte tohoto tlačítka k nalezení cesty

Tlačítko 2. – Vytvoří soubor, do kterého zapíše přeformátovaný TCX soubor

Tlačítko 4. - Vytvoří soubor, do kterého zapíše přeformátovaný TCX soubor včetně zvolené synchronizace A nebo B. V části C lze po vyhledání souboru tlačítkem 1., start pro synchronizace, pokud uživatel nezvolí čas žádný zvolí si start program automaticky.



Obr. 39 Vyobrazení programu s přidaným označením pro funkci návodu (Zdroj: Autor)

4 Ověření funkčností programu

Ověření funkčnosti proběhlo u dvou souborů TCX, prověření zde bylo úspěšné. K těmto souborům nebyly k dispozici hodnoty z mobilní spirometrie, proto jsme využili jiných hodnot spirometrie, které byly upraveny tak, aby otestovaly funkčnost programu. Zkouška proběhla úspěšně. Ověření bohužel muselo proběhnout v omezeném rozsahu. Více souborů nebylo možno naměřit a posléze otestovat, z důvodu vypuknutí pandemie COVID-19 v době plánovaného měření. Lze očekávat funkčnost i u dalších formátování, která nebylo možno provést, minimálně u čistého přeformátování z TCX do CSV. Pokud si uživatel zvolí využít vytvořené možnosti synchronizace, musí dodržet podmínky popsané v kapitole Omezení vytvořeného programu.

Autor práce je připraven poskytnout technickou podporu po mailové žádosti na a programu na emailové adrese jaroslav.vasa@tul.cz. Děkujeme za pochopení.

5 Závěr

Tato bakalářská práce se zabývala převodem fitness dat do tabulkového formátu. Práce měla tři cíle.

První cíl práce bylo vytvoření programu, který převede fitness data TCX (GNSS, výkon, tepová frekvence a další) do tabulkového formátu CSV.

Druhý cíl práce bylo TCX data správně synchronizovat s daty získanými pomocí mobilní sprirometrie. Obou výše zmíněných cílů bylo dosaženo pomocí vytvořeného programu, který je výstupem této práce. Výsledný program má možnost jak prostého převodu TCX dat, tak možnost převodu a synchronizace. Tím bylo docíleno větší využitelnosti programu, než kdyby byl program úzce specializován pouze na převod se synchronizací. Synchronizace v rámci programu je však omezená omezeními popsanými v kapitole **Omezení vytvořeného programu**. Proto byly v kapitole **Další možnosti synchronizace** popsány další možnosti synchronizace bez využití výstupu práce. Program by mohl být lépe optimalizován, ale vzhledem k velikosti formátovaných dat je program vytvořen pro tuto práci dostačující.

Poslední cíl práce bylo ověřit funkčnost programu. Tento cíl nebyl zcela splněn, ověření proběhlo velmi stroze a jen v testovacích podmínkách. Důvody, proč nebylo možné ověření provést je vysvětleno v kapitole Ověření funkčnosti programu.

Seznam použité literatury

[Schéma serializace] [online obrázek] In:WAGNER, Bill a OLPROD. Serializace (C#)- C# Programovací průvodce. MICROSOFT. Oficiální domovská stránka Microsoft [online]. Redmond: Microsoft, 2019, aktualiz. 2020-02-01 [cit. 2020-04-15]. Dostupné z: https://docs.microsoft.com/cs-cz/dotnet/csharp/programming-guide/concepts/serialization/

BERNACIKOVÁ, Martina, 2012. *Fyziologie*. Brno: Masarykova univerzita. ISBN 978-80-210-5841-5. Dostupné také z: http://www.fsps.muni.cz/emuni/data/reader/book-3/Cover.html

BERNACIKOVÁ, Martina. Anaerobní práh z ventilačního ekvivalentu pro kyslík [online]In: BERNACIKOVÁ, Martina, 2012. *Fyziologie*. Brno: Masarykova univerzita. ISBN 978-80-210-5841-5. Dostupné také z: http://www.fsps.muni.cz/emuni/data/reader/book-3/Cover.html

BERNACIKOVÁ, Martina. Laktátový práh [online]In: BERNACIKOVÁ, Martina, 2012. *Fyziologie*. Brno: Masarykova univerzita. ISBN 978-80-210-5841-5. Dostupné také z: http://www.fsps.muni.cz/emuni/data/reader/book-3/Cover.html

BERNACIKOVÁ, Martina. Ventilační práh [online]In: BERNACIKOVÁ, Martina, 2012. *Fyziologie*. Brno: Masarykova univerzita. ISBN 978-80-210-5841-5. Dostupné také z: http://www.fsps.muni.cz/emuni/data/reader/book-3/Cover.html

BULÍKOVÁ, Táňa. 2014. *EKG pre záchranárov nekardiológov.* Praha: Grada. ISBN 978-80-247-5308-9.

COLINBASHBASH. 2012. *Import Excel File to DataSet*. Code Project [online]. Toronto Ontario, M3C 3G8 Canada: CodeProject, 29.5.2012 [cit. 2020-04-15]. Dostupné z: https://www.codeproject.com/Articles/32370/Import-Excel-File-to-DataSet#xx

COMPEK. Cortex MetaMax 3BR2. Compek medical services, s, r, o. COMPEK. COMPEK.CZ – *Compek medical services, s. r. o* [online]. Jičín: COMPEK MEDICAL SERVICES, s.r.o 2010 [cit. 2020-04-15]. http://www.compek.cz/cortex-metamax-3br2.htm

ČÁPKA, David. 2019. *Lekce 1 - Výjimky* [online]. [cit. 2020-04-12]. Dostupné z: https://www.itnetwork.cz/csharp/soubory/c-sharp-tutorial-vyjimky-try-catch-finally

ČÁPKA, David. 2019. L*ekce 3 - Proměnné, typový systém a parsování*. [online]. [cit. 2020-04-12]. Dostupné https://www.itnetwork.cz/csharp/zaklady/c-sharp-tutorial-promenne-typovy-system-a-parsovani

DAVID, Jakub, 2008. *Lékařské přístroje a zařízení* [online]. [cit. 2019-11-15]. Dostupné z: https://sites.google.com/site/lpz2011123/

DOBIÁŠ, Viliam. 2013. *Klinická propedeutika v urgentní medicíně*. Praha: Grada. ISBN 978-80-247-4571-8.

EL-RABBANY, Ahmed. 2006. *Introduction to GNSS: the Global Positioning System*. 2nd ed. Boston, MA: Artech House. ISBN 9781596930162.

GARMIN. Garmin Edge 520 - Garmin Česká republika. GARMIN. *GARMIN.CZ - Garmin Česká republika* [online]. Praha: Garmin Česká republika, 2018 [cit. 2020-04-15]. Dostupné z: https://www.garmin.cz/garmin-edge-520/77967

GARMIN. Garmin Vector3 Double - Garmin Česká republika. GARMIN. *GARMIN.CZ - Garmin Česká republika* [online]. Praha: Garmin Česká republika, 2018 [cit. 2020-04-15]. https://www.garmin.cz/garmin-vector3-double/79028

GARMIN. Garmin Vector3 Double - Garmin Česká republika. GARMIN. *GARMIN.CZ - Garmin Česká republika* [online]. Praha: Garmin Česká republika, 2018 [cit. 2020-04-15]. https://www.garmin.cz/garmin-vector3-double/79028

HABERL, Ralph, 2012. EKG do kapsy. Praha: Grada. ISBN 978-80-247-4192-5.

HAMERNÍK, Pavel. 2016. *Procedurální generování měst*. Brno. Bakalářská práce. Vysoké učení technické v Brně, Fakulta informačních technologii.

CHALOUPKA, Václav a Lubomír ELBL. 2003.*Zátěžové metody v kardiologii*. Praha: Grada. ISBN 80-247-0327-0.

KOLÁŘ, Radim. 2007. *Lékařská diagnostická technika.* Brno: Vysoké učení technické v Brně.

Místa vyhmatání pulzu [obrázek v knize]. In: MIKŠOVÁ, Zdeňka. 2006. Kapitoly z ošetřovatelské péče. Aktualiz. a dopl. vyd. Praha: Grada. Sestra (Grada), s.70. ISBN 80-247-1442-6.

MOUREK, Jindřich. 2012. *Fyziologie: učebnice pro studenty zdravotnických oborů*. 2., dopl. vyd. Praha: Grada. ISBN 978-80-247-3918-2.

NOVOTNÝ, Jan. 2017. *Zátěžové testy ve sportovní medicíně* [online]. [cit. 2020-03-06]. ISBN 978-80-88246-06-0. Dostupné z: https://publi.cz/books/132/index.html?secured=false#cover

NOVOTNÝ, Jan. Příklad stanovení „cirkulačního prahu“. In:NOVOTNÝ, J., M.,SEBERA, M.,NOVOTNÁ, L., HRAZDIRA, A., CHALOUPECKÁ. 2006.Kapitoly sportovní medicíny [online]. Brno: Masarykova Univerzita, [cit.2011-11-18]. Dostupné z WWW: https://is.muni.cz/auth/do/rect/el/estud/fsps/ps06/sportmed/web/index.html.

NOWAKOVÁ, Beata. 2017. *Bezpečnostně technické prohlídky fotopletismografických přístrojů.* Brno: Bakalářská práce. Technická univerzita Ostrava, Fakulta elektrotechniky a informatiky. Katedra kybernetiky a biomedicínského inženýrství.

OSCAR, Jara. Find text in string with C# - Stack Overflow. STACK OVERFLOW. *Stack Overflow - Where Developers Learn, Share, & Build Careers* [online]. New York: Stack Exchange, 2012, aktualiz. 2020-01-21 [cit. 2020-04-15]. Dostupné z: https://stackoverflow.com/questions/10709821/find-text-in-string-with-c-sharp

PASTUCHA, Dalibor. 2014. *Tělovýchovné lékařství: vybrané kapitoly*. Praha: Grada. ISBN 978-80-247-4837-5.

Převodní systém srdeční [obrázek v knize]. In: ELLENBOGEN, Kenneth A, et al. 2017. Clinical cardiac pacing, defibrillation, and resynchronization therapy. 5th ed. Philadelphia: Elsevier, s.69. ISBN 978-0-323-37804-8.

SHAFRANOVICH, Yakov, RFC 4180. *IETF Tools* [online]. 2005 [cit. 2020-01-25]. Dostupné z: https://tools.ietf.org/html/rfc4180#section-7.2

SILBERNAGL, Stefan a Agamemnon DESPOPOULOS. 2016. *Atlas fyziologie člověka*. 4. čes. vyd., překlad 8. německého vydání. Přeložil Otomar KITTNAR, ilustroval Wolf-Rüdiger GAY, ilustroval Astried ROTHENBURGER. Praha: GRADA Publishing. ISBN 978-80-247-4271-7.

SOVOVÁ, Eliška, 2006. *EKG pro sestry*. Praha: Grada. Sestra (Grada). ISBN 80-247-1542-2.

ŠEFLOVÁ, Iva, 2014. *Pohyb a zdraví: inovace výuky tělesné výchovy a sportu na fakultách TUL v rámci konceptu aktivního životního stylu*. Liberec: TUL. ISBN 978-80-7494-122-1.

Thraka a OLPROD. Použití tříd kódování znaků v rozhraní .NET. C# Programovací průvodce. MICROSOFT. *Oficiální domovská stránka Microsoft* [online]. Redmond: Microsoft, 2019, aktualiz. 2017-22-12 [cit. 2020-04-15]. Dostupné z: https://docs.microsoft.com/cs-cz/dotnet/standard/base-types/character-encoding

VONDRUŠKA, Vladimír a Karel BARTÁK. 1999. *Pohybová aktivita ve zdraví a v nemoc*i. Hradec Králové: Klinika tělovýchovného lékařství FN a LFUK. Poradna zdravého životního stylu. ISBN 80-238-4536-5

WAGNER, Bill a OLPROD. Jak převést řetězec na číslo - C# Programovací průvodce. MICROSOFT. *Oficiální domovská stránka Microsoft* [online]. Redmond: Microsoft, 2019, aktualiz. 2019-02-11 [cit. 2020-04-15]. Dostupné z: https://docs.microsoft.com/cs-cz/dotnet/csharp/programming-guide/types/how-to-convert-a-string-to-a-number

WAGNER, Bill a OLPROD. Serializace (C#)- C# Programovací průvodce. MICROSOFT. *Oficiální domovská stránka Microsoft* [online]. Redmond: Microsoft, 2019, aktualiz. 2020-02-01 [cit. 2020-04-15]. Dostupné z: https://docs.microsoft.com/cs-cz/dotnet/csharp/programming-guide/concepts/serialization/

WAGNER, Bill a OLPROD. try-catch (Referenční dokumentace jazyka C#). C# Programovací průvodce. MICROSOFT. *Oficiální domovská stránka Microsoft* [online]. Redmond: Microsoft, 2019, aktualiz. 2015-20-07 [cit. 2020-04-15]. Dostupné z: https://docs.microsoft.com/cs-cz/dotnet/csharp/language-reference/keywords/try-catch

YAHCH. GitHub - yahch/OpenTcx: Garmin Tcx file Analyzor and Generator for .Net. GITHUB. *The world's leading software development platform · GitHub* [online]. San Francisco: GitHub, 2019, aktualiz. 2019-08-13 [cit. 2020-04-15]. Dostupné z: https://github.com/yahch/OpenTcx

Seznam příloh

Příloha A: CD ROM

- Elektronická podoba práce
- Návod k výstupu práce
- Aplikace výstupu práce
- Projekt výstupu práce