Labovi cheat sheet

# GIT

* Pozicionirati se u lokalni folder koji odgovara vašem repozitoriju.
* Desni klik -> Open Git Bash HERE
* Nakon otvaranja Git Basha, iza lokacije (žuta slova) bi trebalo sa plavim slovima pisati (main) ili (master)
* Utipkati „git config –global user.name “vaš GH username“
* Utipkati „git config –global user.email “vaš GH email“
* Utipkati „git fetch“ -> ako ispiše da imate commita koji su na GH, ali ne i u vašem lokalnom repozitoriju -> utipkati „git pull“
* Kad napravite neke promjene i sve ih spremite u Git Bashu utipkajte „git status“, tu će vam se ispisati imate li kakve promjene -> ako imate utipkajte „git add .“ tako da stageate sve vaše promjene -> utipkajte „git commit -m “opis vaših promjena““ (NAPOMENA: ako zaboravite stavit -m i/ili “opis vaših promjena“ postoji šansa da vam se upali VIM Editor u kojeg možete upisati vaš commit, a iz editora se izlazi naredbom :q!) -> utipkajte „git push“ i git će sve vaše promjene poslati na vaš GH i tamo će te ih moći vidjeti i na nekom drugom PC-u ih „git pull“.
* NAPOMENA ako vam u nekom trenutku git vrati grešku 403 -> otvorite Start meni -> utipkajte „Credential Manager“ -> otvorite program -> kliknite na „Windows Credentials“ -> pod „Generic Credentials“ nađite „git:https://github.com“ -> Remove
* NAKON SVAKE LABORATORIJSKE VJEŽBE SE NE ZABORAVITE ODLOGIRATI IZ VAŠEG GH ACCOUNTA U BROWSERU

# VISUAL STUDIO - NOVI PROJEKT

* Upaliti Visual Studio po želji (može bit 2017, 2022... svejedno, samo da nije Visual Studio Code)
* File -> New -> Create a new project
* C++ (Visual C++) -> Empty Project
* Project name (Nešto što označava koji zadatak/lab/vježbu odrađujete u tom projektu) -> Location BROWSE (OBAVEZNO mora biti odabran vaš lokalni folder koji je povezan sa vašim GH repozitorijem) -> Create
* U „Solution Exploreru“ (upaliti ako ga nema na ekranu) -> desni klik na „Source Files“ -> Add -> New Item... -> „Neki naziv.c“ (bitno je staviti ekstenziju .c)
* Pisati sav kod u tome novostvorenom fileu

# DOBRE PRAKSE I SAVIJETI

* Jednolinijski komentari se pišu iza // -> (//ovo je jednolinijski komentar)
* Više linijski komentari se pišu između /\* i \*/ ->

(/\* ovo je

više linijski

komentar \*/)

* Za komentirati blok koda koristiti kraticu (CTRL + K + C), a za od komentirati (CTRL + K + U)
* OBAVEZNO iznad svakog dvosmislenog komada koda napisati komentar što taj komad koda radi
* Dodati tekst zadatka u komentar na vrhu filea u kojem programirate
* OBAVEZNO pisati deskriptivne nazive varijabli, funkcija, strukturi... (int addToEndOfList(Position head);)
* OBAVEZNO svaku definiranu varijablu inicijalizirati na neku vrijednost (int birthYear = 0;)
* Generalno za manje više sve koristiti „cammelCase“ način pisanja imena (iznimka su #define-ovi)
* Sve pisati na engleski jer će te tako i u poslovnom životu raditi
* OBAVEZNO Deklarirati funkcije sa njihovim potpisom iznad main() pa onda ih definirati ispod main()   
  (int addToEndOfList(Position head);  
  ...  
  int main() { ... }  
  ...  
  int AddToEndOfList(Position head) { ... }  
  ...)
* OBAVEZNO koristiti return mogućnost funkcija (nema void funkcija)
  + Ako je neka funkcija koja prebrojava nešto vratiti pozitivan broj (prebrojani broj) kao znak da je ispravno završila, vratiti 0 ako nije ništa prebrojala, a vratiti negativan broj ako je bila neka greška u funkciji
  + Ako je neka funkcija koja inače ne vraća vrijednost, vratiti 0 kad se sve uredu izvrši, vratiti pozitivan broj ako je neka korisnička greška (recimo neki krivi unos), a negativan broj ako je neka sistemska greška (recimo greška sa mallocom ili otvaranjem file)
* OBAVEZNO definirati greške i njihove returnove na vrhu filea (#define FILE\_NOT\_OPENED (-1))
* OBAVEZNO definirati „magic numbers“ na vrhu filea (#define MAX\_LINE (1024))
* OBAVEZNO iza većine poziva funkcija provjeriti što je funkcija vratila i obraditi grešku ili nastaviti dalje sa kodom
* OBAVEZNO odmah nakon korištenja fopen() funkcije napisati fclose() funkciju da ne bi ste zaboravili kasnije.
* OBAVEZNO čim prestanete koristiti vrijednosti za koje ste dinamički alocirali memoriju (malloc()) oslobodite tu memoriju (free())
* OBAVEZNO ne prosljeđivati (FILE\* filePointer) kao parametar/argument funkcijama nego u svakoj funkciji zasebno otvarati i zatvarati file
* U prvoj liniji filea definirati (#define \_CRT\_SECURE\_NO\_WARNINGS)
* Strukture i pokazivače na njih definirati na ovaj način  
  (struct \_Person;

typedef struct \_Person\* Position;

typedef struct \_Person

{

char name[MAX\_LENGTH];

char surname[MAX\_LENGTH];

int birthYear;

Position next;

} Person;)