Zadatci za 1. laboratorijske vježbe ASP 2018./2019.

1. Napišite funkciju čiji je prototip:

   void ispis (float polje[], int n) ;

   koja kao argumente prima polje **polje**(tj. pokazivač na početak polja), čiji su elementi tipa **float** i broj članova polja (**n**) te rekurzivno ispisuje negativne članove polja od prvoga prema zadnjemu.

   Primjer: za polje [0, -1.2, 2.5, 3.1, -4.17, 5.19, -6.91] treba biti ispisano: -1.2, -4.17, -6.91.

   Napišite glavni program koji će učitati broj elemenata jednodimenzionalnog polja **n** i alocirati polje **A** od **n** članova tipa **float** (možete koristiti operator **new** ili funkciju **malloc**). Nakon toga učitajte **n** elemenata polja **A**. Negativne članove polja ispišite od prvoga prema zadnjemu korištenjem funkcije **ispis**.

2. Napišite funkciju čiji je prototip:

   int zbrojiKvadrate (int polje[], int n) ;

   koja kao argumente prima polje **polje** i broj članova (**n**) te rekurzivno zbraja članove polja koji su kvadrati nekog drugog prirodnog broja.
   Primjer: za polje [0, 1, 2, 3, 4] funkcija treba vratiti 5 (zbrojeni su 1 i 4, koji su kvadrati brojeva 1 i 2).

   Napišite glavni program koji će učitati broj elemenata jednodimenzionalnog polja **n** i stvoriti cjelobrojno polje **A** od **n** članova (možete koristiti operator **new** ili funkciju **malloc**).
   Potrebno je napuniti polje **A** s **n** slučajno odabranih prirodnih brojeva iz intervala [1, 100] te ispisati polje i zbroj članova polja koji su kvadrati nekog drugog prirodnog broja (pozvati funkciju **zbrojiKvadrate**).

3. Napišite funkciju čiji je prototip:

   double pi (int n) ;

   koja kao argument prima broj članova reda (**n**) te rekurzivno računa broj π prema izrazu:

   $\frac{\pi}{4} = 1 - \frac{1}{3} + \frac{1}{5} - \frac{1}{7} + \frac{1}{9} - \cdots$

   Primjer: za n = 1, funkcija treba vratiti 4; za n = 2, funkcija treba vratiti 2.666667; za n = 10, funkcija treba vratiti 3.041840, itd.

   Napišite glavni program koji će učitati broj elemenata jednodimenzionalnog polja **n** i stvoriti polje **A** od **n** članova tipa **double** (možete koristiti operator **new** ili funkciju **malloc**). Zatim polje treba popuniti tako da član polja **A[i]** sadrži aproksimaciju broja π izračunatu korištenjem funkcije **pi** za **i+1** članova reda. Npr. za **n** = 3, polje treba sadržavati vrijednosti [4.000000, 2.666667, 3.466667]. Ispisati članove polja **A**.

4. Napišite funkciju čiji je prototip:

   double exp (double x, int n, int *fakt, double *xpot);

   koja kao argument prima realni broj **x** i broj članova reda (**n**) te rekurzivno računa e^x prema izrazu:

$$e^x = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \frac{x^4}{4!} + \cdots$$

Funkcija prima i dva dodatna argumenta: **fakt** (adresa na kojoj je pohranjen **n!** za trenutni **n**) i **xpot** (adresa na kojoj je pohranjena **n**-ta potencija broja **x** za trenutni **n**). Korištenjem ovih pomoćnih argumenata trebate napisati funkciju **exp** koja se izvodi u vremenu **O(n)**.

Primjer: za x = 1 i n = 0, funkcija treba vratiti 1; za x = 1 i n = 1, funkcija treba vratiti 2; za x = 1 i n = 3 funkcija treba vratiti 2.666667; x = 1 i n = 10, funkcija treba vratiti 2.718282, itd.

Napišite glavni program koji će učitati broj elemenata jednodimenzionalnog polja **n** i broj **x** tipa **double**. Stvorite polje **A** od **n** članova tipa **double** (možete koristiti operator **new** ili funkciju **malloc**). Zatim polje treba popuniti tako da član polja **A[i]** sadrži vrijednost izraza $\mathbf{e^x}$ izračunatog korištenjem funkcije **exp** za **i** članova reda. Npr. za **n** = 5 i **x** = 1, polje **A** treba sadržavati vrijednosti [1.000000, 2.000000, 2.500000, 2.666667, 2.708333]. Ispisati članove polja **A**.

5. Napišite funkciju čiji je prototip:

   template <typename T> int binarnoTrazi (T polje[], int n, T x) ;

   koja kao argumente prima pokazivač na početak uzlazno sortiranog polja (**polje**) čiji su članovi tipa **T**, broj članova polja (**n**) te broj **x**. U funkciji postupkom binarnog pretraživanja treba provjeriti nalazi li se **x** u polju. Funkcija vraća indeks elementa **x**, ako se **x** nalazi u polju, a -1 inače.

   Napišite glavni program koji će učitati broj elemenata jednodimenzionalnog polja **n** te realni broj **x**.

   Stvorite polje **A** od **n** članova tipa **float**(možete koristiti operator **new** ili funkciju **malloc**).
   Potrebno je napuniti polje **A** s **n** vrijednosti tako da je **A[i]** = i * 1.1 te ispisati članove polja. Za broj **x** potrebno je provjeriti nalazi li se u polju **A** (koristite funkciju **binarnoTrazi**). Potrebno je ispisati indeks člana polja, ako je **x** pronađen u polju **A**, a poruku „Broj se ne nalazi u polju.“, ako **x** nije pronađen u polju **A**.

   Ponovite postupak s članovima tipa **int**, tako da je **A[i]** = i + 3;

6. Napišite funkciju čiji je prototip:

   char *ostaviSlova (string ulaz);

   koja kao argument prima std::string **ulaz**, a vraća pokazivač na početak novog **znakovnog niza** za koji je dinamički alocirana memorija u funkciji (možete koristiti operator **new** ili funkciju **malloc**). Novi niz treba sadržavati samo znakove engleske abecede u istom poretku kao u ulaznom stringu.

   Primjer: za ulazni string "asp12_i_ASP13", funkcija treba vratiti pokazivač na novi niz "aspiASP“.

   Napišite glavni program u kojem će biti definiran string (varijabla **ulaz)** sadržaja "asp12_i_ASP13" te ispisati znakovni niz koji je rezultat poziva funkcije **ostaviSlova** s argumentom **niz**.

7. Napišite funkciju koja prima pokazivač na polje cijelih brojeva i koja vraća pokazivač na novo polje koje se sastoji od nasumično poredanih kvadriranih elemenata ulaznog polja.
   Primjerice, ako se ulazno polje sastoji od elemenata 1, 2, 3, 4 i 5, izlazno polje može biti 25, 16, 1, 9, 4.

Potrebno je napisati i glavni program koji od korisnika učitava cijeli broj **n** te zatim rezervira memorijski prostor za cjelobrojno polje (**polje**) od **n** članova (možete koristiti operator **new** ili funkciju **malloc**). Članove polja **polje** treba učitati s tipkovnice. Glavni program zatim poziva funkciju i ispisuje rezultat izvođenja funkcije (izlazno polje).

8. Napišite razred **SanitizedString** koji sadrži privatnu varijablu **str** tipa **std::string** (ili **char***), i public metode **removeDuplicateWhitespace** i **removeNonAplhaChars**.
   Metoda **removeDuplicateWhitespace** modificira **str** tako da iz ulaznog stringa izbacuje sve pojave višestrukih praznina. Primjerice, za string „Sunce nam dolazi!“, metoda će ga prepraviti u niz „Sunce nam dolazi!“ (umjesto višestrukih praznina je ostala samo po jedna praznina između riječi). Metoda **removeNonAplhaChars** izbacuje sve znakove koji nisu slova abecede (dovoljno je da program radi samo za znakove engleske abecede). Npr. ako je **str** = „M~ir4ko&“, funkcija ga treba prepraviti u „Mirko“.
   Po potrebi napisati dodatne metode koje trebaju za ostvarenje funkcionalnosti (konstruktore, destruktor, gettere, settere…)
   Razred treba omogućiti i ispis sanitiziranih stringova pomoću operatora <<.

   Potrebno je napisati glavni program koji od korisnika učitava string (ili znakovni niz). Program zatim stvara objekt tipa **SanitizedString** i ispisuje početni i sanitizirani string.

9. Napišite **rekurzivnu** funkciju koja kao parametar prima polje cijelih brojeva i njegovu veličinu. Prototip funkcije je:

   void f (int polje[], int n, int m);

   Funkcija treba polje popuniti rastućim vrijednostima koje su potencije broja m, na način da element na indeksu 0 ima vrijednost m^0, na indeksu 1 vrijednost m^1 itd. Elementi polja idu do m^(n-1).

   Potrebno je napisati i glavni program koji od korisnika učitava cijele brojeve **n i m** te zatim stvara cjelobrojno polje od **n** članova(možete koristiti operator **new** ili funkciju **malloc**). Glavni program zatim poziva funkciju **f** za to cjelobrojno polje, broj članova polja **n** i parametar **m** te ispisuje rezultate izvođenja funkcije.

10. Napišite **rekurzivnu** funkciju čiji je prototip:

    double f (double z, int k);

    koja kao argument prima realni broj **z** i cijeli broj **k** te u vremenu O(k) računa izraz:

    $$\frac{(-1)^k z^{2k+1}}{(2k+1)!}$$

    U glavnom programu je potrebno definirati realni broj **z** = 0.5 te funkciju **f** pozivati za različite vrijednosti broja **k** i ispisivati rezultate poziva.