Smijete koristiti pisane materijale ili vlastite bilješke spremljene u HOME direktoriju, web-stranicu kolegija i linkove koji vode s nje.

Strogo je zabranjeno korištenje ChatGPT-a i ostalih generativnih modela te će se isto provjeravati za svaki kod. Ne smijete se koristiti nikakvim sredstvima komunikacije (papirići, mobitel, mail, chat, messenger i sl.). Svaki pokušaj prepisivanja i dogovaranja rezultirat će trenutnim udaljavanjem sa kolokvija. Strogo je zabranjeno fotografiranje ili pretipkavanje teksta zadataka.

Iznimno, studenti koji kolokvij pišu u čitaonici, smiju upaliti hotspot na mobitelu, nakon čega mobitel moraju okrenuti licem prema stolu na prikladnoj udaljenosti. Mobitel se nakon toga više ne smije dirati do kraja izvođenja kolokvija.

Prvi zadatak (15 bodova)

Implementirajte sljedeću hijerarhiju klasa:

- Apstraktna klasa `Popis` predstavlja proizvoljan popis (npr. popis za kupovinu, popis zadataka, popis knjiga za pročitati,...). Ova klasa treba imati člansku varijablu `std::string imeVlasnika` (ime vlasnika popisa).
- Klasa `PopisZaKupovinu` nasljeđuje klasu `Popis`, a ima dodatnu člansku varijablu `std::map<std::string, int> artikli`. Ključevi preslikavanja su artikli koji se nalaze na popisu za kupovinu, a vrijednosti preslikavanja predstavljaju količinu artikla, tj. koliko komada pojedinog artikla je potrebno kupiti.
- Klasa `PopisZadataka` nasljeđuje klasu `Popis` te predstavlja popis zadataka koje osoba mora obaviti u jednom danu (tzv. *to do* lista). Ima dodatne članske varijable `std::string datum` te `std::map<int, std::string> zadaci`. Ključevi preslikavanja predstavljaju vrijeme (puni sat), a vrijednosti predstavljaju zadatak koji u to vrijeme treba obaviti.

Implementirajte konstruktore (sa i bez parametara) i destruktore za sve tri klase. Konstruktori s parametrima primaju sve članske varijable. Destruktori ispisuju `Unisten <klasa>.\n`.

Implementirajte člansku funkciju `void dodajStavku(std::string stavka, int info)` koja dodaje stavku na popis prema sljedećim pravilima:

- klasa `Popis`: čista virtualna funkcija
- klasa `PopisZaKupovinu`: parametar `stavka` predstavlja artikl, a parametar `info` predstavlja količinu artikla. Ako artikli `stavka` već postoji na popisu, količina tog artikla poveća se za `info`.
- klasa `PopisZadataka`: parametar `stavka` predstavlja zadatak, a parametar `info` predstavlja vrijeme u koje je zadatak potrebno izvršiti. Ako u zadano vrijeme već postoji neki zadatak na popisu, taj se zadatak zamijenjuje zadatkom `stavka`.

Implementirajte člansku funkciju `void ispisi()` koja ispisuje na ekran stanje svih članskih varijabli za taj objekt. Sami odaberite format ispisa.

Implementirajte globalnu funkciju `PopisZaKupovinu spojiPopiseZaKupovinu()` koja sve popise za kupovinu koji se nalaze u glavnom programu spaja u jedan popis za kupovinu. Ime vlasnika novog popisa sastoji se od imena vlasnika svih popisa za kupovinu u glavnom programu odvojenih zarezom.

Napomene:

- Svoje rješenje spremite u jednu datoteku `popis.cpp`.
- Sve varijable u klasama moraju biti `private`. Smijete implementirati i dodatne članske funkcije i varijable, no i dalje sve članske varijable trebaju biti zaštićene. Zabranjeno je korištenje globalnih varijabli i `friend` funkcija i klasa.
- Izbjegavajte dupliciranje dijelova koda; u protivnom možete izgubiti bodove.
- Osigurajte da se pozovu svi potrebni destruktori prilikom uništavanja izvedenih klasa. Pobrinite se da se oslobodi sva dinamički alocirana memorija.
- Datoteka `popis-main.cpp` koja sadrži *main* funkciju za testiranje dostupna je na Merlinu.
- Zabranjeno je korištenje ključne riječi `auto`.

Drugi zadatak (20 bodova)

Implementirajte klasu `Polinom` koja sadrži privatne članove:

- `vector<double> a`: vektor koeficijenta polinoma, gdje je `a[i]` koeficijent uz $x^{\mathtt{i}}$.
- `void skrati()`: funkcija koja briše vodeće nule iz `a`, osim koeficijenta uz x^0.

Implementirajte sljedeće javne funkcije:

- `Polinom()`: stvara nul-polinom.
- `Polinom(vector<double> v)`: stvara polinom sa danim vektorom koeficijenta.
- `unsigned stupanj()`: vraća stupanj polinoma.

Dodatno, omogućite rad sljedećih operatora:

- `operator<<`: ispisuje polinom u obliku `a[n]*x^n + a[n-1]*x^n-1 + ... + a[1]*x + a[0]`, gdje je `n` stupanj polinoma („...” se zamjenjuje s ostalim monomima).
- `operator[]`: prima `unsigned i` i vraća koeficijent uz $x^{\mathtt{i}}$ (ako je `i` strogo veći od stupnja polinoma, vraća nulu).
- `operator+=`: zbrajanje polinoma.
- `operator/=`: dijeljenje polinoma skalarom.
- `operator*`: množenje polinoma skalarom ili množenje dva polinoma.

Implementirajte funktor `SkalarniProdukt`, gdje `operator()` prima reference na dva `Polinoma` `p` i `q` te vraća $\int_{-1}^{1} \mathtt{p}(x)\mathtt{q}(x)\,\mathrm{d}x$.

Implementirajte generičku funkciju `gram_schmidt` koja prima iteratore `st` i `en` na redom početak i kraj spremnika (koji sadrži objekte koji implementiraju gornje aritmetičke operatore) te binarni funktor `f` (koji prima dva objekta i vraća `double`). Funkcija nad elementima unutar intervala $[\mathtt{st}, \mathtt{en}\rangle$ primjenjuje Gram-Schmidtov algoritam sa `f` kao skalarnim produktom:

Ulaz: Vektori $v_1, \ldots, v_n$ i skalarni produkt $\langle \cdot, \cdot \rangle$.

za $i = 1, \ldots, n$
  za $j = 1, \ldots, i-1$
    $v_i = v_i - \langle v_i, v_j \rangle v_j$
  $v_i = v_i / \sqrt{\langle v_i, v_i \rangle}$

Implementirajte globalnu funkciju `vector<Polinom> polinomi()` koja vraća vektor svih `Polinoma` u glavnom programu.

Napomene:

- Rješenje spremite u datoteku `polinom.cpp`.

IME I PREZIME: JMBAG: BODOVI:

- Zabranjeno je korištenje globalnih varijabli i funkcija iz zaglavlja `algorithm`.
- Smijete implementirati dodatne članske varijable, no i dalje sve članske varijable moraju biti privatne. Smijete implementirati dodatne funkcije (članske i globalne).
- Osigurajte da ne dolazi do kopiranja memorije prilikom prijenosa parametara funkcijama i operatorima (osim ako to nije nužno).
- Onemogućite mijenjanje svih objekata proslijeđenih funkcijama i operatorima ukoliko njihovo mijenjanje nije nužno za implementaciju navedenih funkcionalnosti.
- Datoteka `polinom-main.cpp` koja sadrži `main` funkciju za testiranje dostupna je na Merlinu.