Plenum: Beholdere og generiske klasser 1

Ahmed Waseem Saeed

Februar 2024

Introduksjon

Velkommen til ny plenumstime! Denne uken skal vi bevege oss fra arv til beholdere og generiske klasser. Dette er temaet for to uker, og koden vi begynner på denne uken, skal videreføres til neste uke. Vi skal implementere en association list-datastruktur, som er en lenkeliste med nøkkel og verdi lagret i hver enkelt node. Se denne wiki-artikkelen for mer info om datastrukturen. Merk at oppgaveteksten ikke er like utfyllende denne uken, ettersom det blir en del tegning og forklaring i selve plenumstimen. For å teste koden er filen TestAListe.java lagt ved i samme mappe som denne oppgaveteksten. Vi anbefaler at du tester koden underveis (også med et eget test-program), slik at du lett kan fange opp feilen der den oppstår, fremfor å teste alt på samme gang. Merk at plenumsinnholdet her og neste uke bygger sterkt på de to foregående årene i IN1010.

Tips

TEGN! Både for innleveringen, denne timen, samt oppgaver med innsetting og fjerning i lister (der du må holde styr på pekere), så er det kritisk å kunne visualisere ulike scenarioer og hvordan koden skal håndtere disse. Derfor anbefaler vi på det sterkeste at tegner ved siden av og før implementeringen av metodene under.

Du står i tillegg fritt til å skrive hjelpemetoder dersom du ønsker det.

AListe

Klassen AListe skal representere vår implementasjon av association list. Klassen må inkludere typeparametere (eng: generics eller type parameters) K (key) og V (value), som skal brukes i Node-klassen vår.

Skriv den indre klassen Node, som har en referanse til en forrige- og neste-peker. Klassen må i tillegg ha en variabel av type K som kalles nokkel, samt en

av type V, som heter verdi.

Selve klassen trenger kun tre instansvariabler: start og slutt (begge av typen Node), samt en heltallsvariabel storrelse, som angir antall elementer i lista vår (denne må oppdateres ved innlegging og fjerning av elementer).

hentStorrelse

Metoden hentStorrelse skal returnere størrelsen til lista (antall elementer).

leggTil

Metoden leggTil skal ta inn to argumenter (K nokkel, og V verdi) og legge til en ny node i listen. Dersom nøkkelen allerede eksisterer så skal metoden erstatte verdien med det gitte argumentet.

toString

Allerede på dette stadiet kan du overskrive toString-metoden til klassen. Dette vil la oss teste koden lettere, ved å se hvilke elementer som eksisterer i klassen. Benytt en hjelpepeker til å traversere listen og bygg opp en streng som viser innholdet i listen (med både nøkkel og verdi for hver node).

Testprogram

Skriv et testprogram som oppretter en AListe med type

$$< String, String >$$

Legg til noen elementer og skriv ut listen. Er listen bygget opp slik den var tiltenkt? Hvis ikke, gå tilbake og identifiser feilene.

hent

Metoden hent har en parameter, nokkel, og skal traversere listen helt til den finner en node med gitt nøkkel. Dersom nøkkelen eksisterer i lista, skal metoden returnere verdien. Hvis ikke, skal metoden returnere null.

fjern

Metoden skal kunne ta inn en nøkkel og fjerne noden fra listen. Merk at dette er en vrien operasjon som må bevare rekkefølgen i listen, og oppdatere pekerne korrekt. Tegn gjerne før du begynner. Metoden skal returnere den fjernede verdien, evt. null dersom nøkkelen ikke eksisterer eller lista er tom.