Dinamički životni vek

- Dinamički životni vek objekata neposredno se kontroliše logikom i dinamikom programa: dinamički objekti se prave i uništavaju eksplicitno:
  - svako izvršavanje izraza (operatora) *new* pravi nov dinamički objekat
  - tako napravljen dinamički objekat živi dok se ne uništi operatorom *delete*

```
Clock* pc = new Clock(9,0,0);
...
delete pc;
```

Pravljenje dinamičkog objekta

Uništavanje dinamičkog objekta

- Životni vek dinamičkih objekata nije implicitan i vezan za neki opseg, kao što je to slučaj sa automatskim i statičkim objektima; on se kontroliše eksplicitno, pa dinamički objekti mogu da nadžive izvršavanje funkcije u kojoj su napravljeni, što tipično i jeste slučaj: oni se prave u jednom scenariju, izvršavanjem jedne funkcije, a uništavaju možda u nekom poptuno drugom scenariju i u drugoj funkciji:

```
template <typename T>
List<T>& List<T>::addBack (T t) {
  ListElem<T>* e = new ListElem<T>(t, tail, nullptr);
  if (!head) head = e;
  tail = e;
  return *this;
}

template <typename T>
T List<T>::removeFront () {
  ListElem<T>* e = head;
  ...
  delete e;
  ...
}
```

Pravljenje dinamičkog objekta

Uništavanje dinamičkog objekta

Dinamički životni vek

- Izraz (operator) *new* uvek radi sledeće stvari, ovim redom:
    1. alocira prostor za smeštanje jednog objekta datog tipa ili niza objekata datog tipa, ako se pravi niz
    2. inicijalizuje jedan objekat ili svaki objekat u nizu (ako se pravi niz objekata) zadatim inicijalizatorom
    3. vraća vrednost tipa pokazivača na dati tip koji ukazuje na napravljeni objekat, ili na prvi element napravljenog niza (ako se pravi niz objekata datog tipa)
- Alokacija prostora (prvi korak) vrši se pozivom neke od (preklopljenih) operatorskih funkcija koje su standardno definisane jezikom (postoje i ovakve operatorske funkcije sa još nekim parametrima):

```
void* operator new   (std::size_t count);
void* operator new[] (std::size_t count);
```

- Tip *size_t* je neoznačeni celobrojni tip deklarisan u nekoliko zaglavlja standardne biblioteke koji se koristi za izražavanje veličina tipova; ovo je tip rezultata operatora *sizeof*
- Parametar ovih funkcija tipa *size_t* prenosi veličinu prostora koji treba alocirati (u jedinicama *sizeof(char)*)
- Podrazumevana implementacija ovih funkcija upravlja slobodnom memorijom (*heap*, *free store*), odnosno radi dinamičku alokaciju i dealokaciju u za to predviđenom delu memorije programa
- Programer može promeniti način alokacije prostora na neki od sledećih načina:
    - zameniti implementaciju neke od ovih funkcija svojom implementacijom, čime će promeniti način alokacije dinamičke memorije (preusmeriti na svoj alokator); dovoljno je samo negde u programu definisati takvu funkciju, bez posebne deklaracije na nekom drugom mestu
    - definisati ove funkcije za svoje klasne tipove, čime će promeniti način alokacije prostora samo za objekte te klase