ALGEBRA

PROGRAMIRANJE

Predavanje 12 – Hrpa

Ishod učenja 5

1

KORIŠTENJE DINAMIČKE MEMORIJE

ALGEBRA

2

Organizacija memorije

- Dva glavna dijela memorije svakog procesa su:
  - **Stog** (engl. *stack*)
    - Predefirana veličina je 1 MB
    - Neprikladno za smještanje veće količine podataka
  - **Hrpa** (engl. *heap*)
    - Također se zove i dinamička memorija ili *free store*
    - Veličinu određuje je li aplikacija 32-bitna ili 64-bitna
- Kad izrađujemo varijablu, polje ili objekt, možemo birati želimo li ih kreirati na stogu ili na hrpi:
  - Ako radimo kao do sada, kreiramo ih na stogu
  - Ako ih želimo kreirati na hrpi, treba nam posebna sintaksa

ALGEBRA

Korištenje dinamičke memorije

- Korištenje dinamičke memorije se radi u tri koraka:
  1. Alociranje (uzimanje, rezerviranje) dijela dinamičke memorije
  2. Korištenje dinamičke memorije
  3. Dealociranje (otpuštanje) dinamičke memorije (kad nam više ne treba)

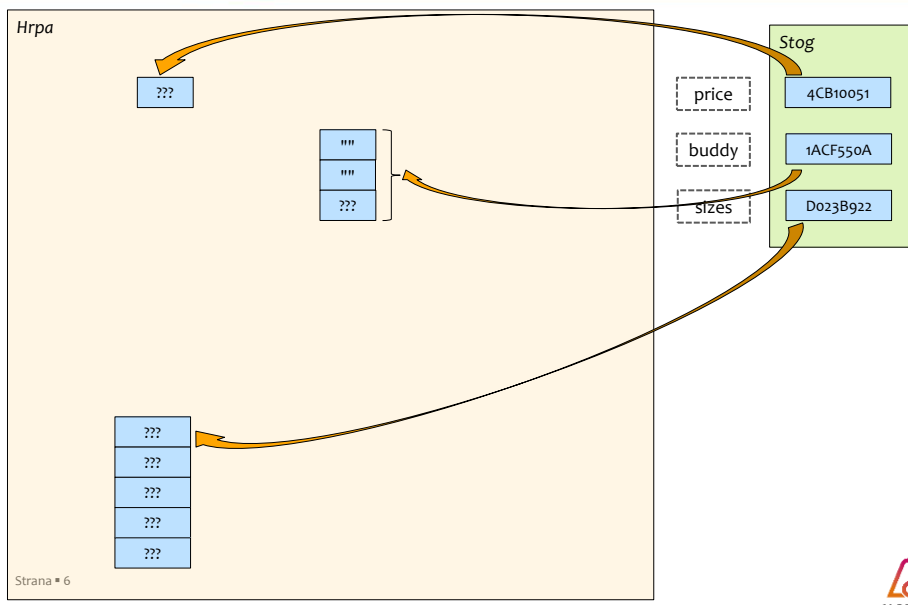
ALGEBRA

Alociranje dinamičke memorije

- Kad nešto želimo spremiti na hrpu, koristimo operatore **new** i **new[ ]**
  - Operator new alocira varijablu ili objekt na hrpi
  - Operator new[ ] alocira polje na hrpi
    - Naziva se dinamičko polje (polje na stogu se naziva statičko polje)
    - Veličina napisana u zaradama može biti konstanta ili **varijabla**!
- Oba operatora vraćaju pokazivač na početak dodijeljenog memorijskog bloka:

```
double* price = new double; // Pokazivač na varijablu
Person* buddy = new Person; // Pokazivač na objekt
int* sizes = new int[5]; // Pokazivač (naziv polja)
```


Vizualizacija



Korištenje dinamičke varijable

- Samo dereferenciramo pokazivač:

```
double* price = new double;

*price = 8.99;

cout << *price << endl;
```

- Dinamičke varijable se rijetko koriste

7

Korištenje dinamičkog objekta

- Dereferenciramo pokazivač i onda pristupimo članu:

```
Person* buddy = new Person;
buddy->first_name = "Izzy";
buddy->last_name = "Walker";
buddy->age = 29;

cout << buddy->first_name << endl;
cout << buddy->last_name << endl;
cout << buddy->age << endl;
```

- Često se koristi za veće objekte

8

Korištenje dinamičkog polja

- Koristi se kao i statičko polje:
  - Pokazivač je u stvari naziv polja
  - Možemo koristiti uglate zagrade ili dereferenciranje pokazivača

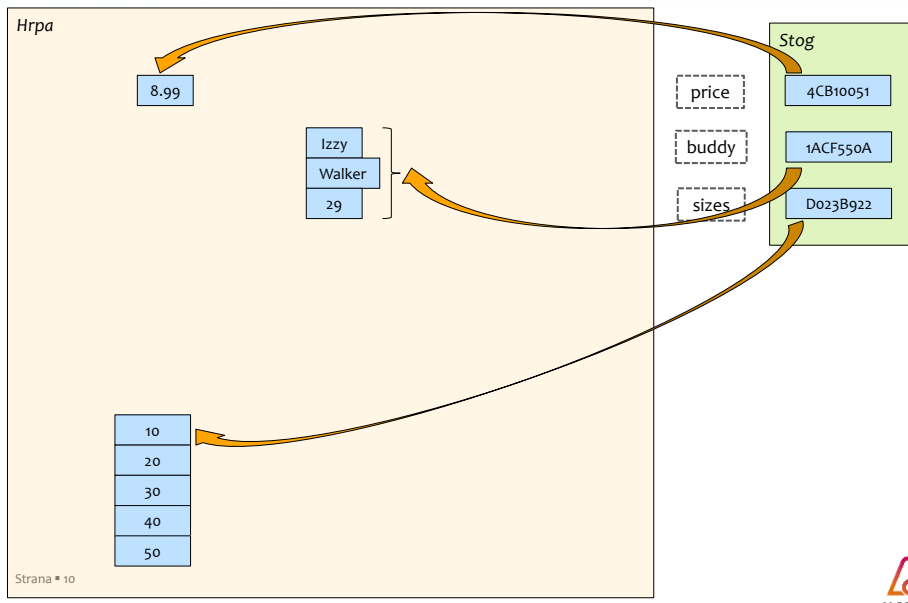
```
int n = 5;
int* sizes = new int[n]; // Možemo koristiti varijablu!

for (int i = 0; i < n; i++) {
    sizes[i] = (i + 1) * 10; // Koristimo uglate zagrade
}

for (int i = 0; i < n; i++) {
    cout << *(sizes + i) << " "; // ili koristimo deref. pokazivač
}
```

- Puno korisnije nego statička polja

ALGEBRA

9

Vizualizacija



ALGEBRA

10

Dealociranje dinamičke memorije

- Čim nam više ne treba, moramo otpustiti dinamičku memoriju da spriječimo gubitak memorije (engl. *memory leak*)
  - Dealociranjem memorija postaje dostupna drugim aplikacijama
- Operatori **`delete`** i **`delete[]`** dealociraju memoriju

```
delete price;
delete buddy;
delete[] sizes;
```

Prazne uglate zagrade!

- Pravila:
  - Ako smo uzeli s `new`, moramo otpustiti s `delete`
  - Ako smo uzeli s `new[]`, moramo otpustiti s `delete[]`
- Nakon otpuštanja ne smijemo više koristiti pokazivač!

ALGEBRA

Primjeri

1. Napravite dvije int varijable na stogu i jednu na hrpi, dodijelite im vrijednost i ispišite ih. Ispišite i adrese na kojoj se varijable nalaze.
2. Napišite program koji u `while` petlji uzima po 4 cijela broja s hrpe i zaboravlja ih osloboditi. Pogledajte kako pada količina slobodne memorije na sustavu.
3. Pitajte korisnika koliko brojeva želi učitati te ih učitajte na hrpu. Nakon toga izračunajte i ispišite njihov zbroj.
4. Pitajte korisnika koliko imena želi učitati te ih nakon toga učitajte. Nakon toga ispišite sva učitana imena tako da svako ime funkcijom ispišete od kraja prema početku.

ALGEBRA

Primjeri

5. Pitajte korisnika koliko brojeva želi učitati te ih nakon toga učitajte. Ako je korisnik unio barem jedan negativan broj, ispišite "GRESKA", inače ispišite "OK".
6. Pitajte korisnika koliko brojeva želi učitati te ih nakon toga učitajte. Nakon toga ispišite jesu li uneseni brojevi rastući (da je svaki sljedeći veći ili jednak prethodnom).
7. Učitajte od korisnika 5 brojeva u obično polje. Nakon toga prebrojite koliko je parnih brojeva uneseno. Nakon toga od tih parnih brojeva napravite novo dinamičko polje i zatim ispišite njegov sadržaj.

13

Primjeri

8. Napravite strukturu za čuvanje podataka o točkama (x, y). Pitajte korisnika koliko točaka želi učitati pa ih učitajte. Ispišite udaljenosti prve točke do svih ostalih (udaljenost d = $\sqrt{(x_2 - x_1)^2+(y_2 - y_1)^2}$)
9. Pitajte korisnika koliko slova želi učitati pa ih učitajte. Ispravna slova su "a", "b" i "c", a ako korisnik unese neko drugo slovo, treba ponoviti unos. Nakon toga ispišite koliko je učitano pojedinih slova, primjerice:

   Slovo a: 3
   Slovo b: 0
   Slovo c: 6

14

Primjeri

10. Napravite strukturu za čuvanje podataka o pravokutnicima (a, b). Pitajte korisnika koliko pravokutnika želi učitati pa ih učitajte. Na kraju ispišite najmanju i najveću površinu pravokutnika.
11. Pitajte korisnika koliko stringova želi učitati pa ih učitajte. Nakon toga ispišite sve stringove koji u sebi sadrže slovo "a". Provjeru sadrži li string slovo "a" radite u funkciji.

ALGEBRA

15

Zadaci za sljedećih 7 dana

- Prije sljedećeg predavanja trebate:
  1. Pročitati iz *Demistificirani C++*:
     - 6.4 Dinamička alokacija objekata
  2. Pogledati sljedeće:
     - W12-1 Dynamic Memory
       - https://youtu.be/ovOvMoOMDAo
     - W12-2 Solving Problems
       - https://youtu.be/HYLPtWw1i6I

ALGEBRA

16