Zavod za elektroniku, mikroelektroniku
računalne i inteligentne sustave

13. travnja 2021.

Oblikovni obrasci u programiranju

2. laboratorijska vježba

1. Razmatramo razvoj podrške za rad s poljima podataka proizvoljnog tipa u programskom jeziku C. Svako polje je složeno u kontinuiranom memorijskom prostoru, a svi podatci u polju imaju isti tip. Polja mogu imati različitu duljinu. Slično kao i u znakovnim nizovima u C-u, duljina polja određena je specijalnim podatkom kojeg nazivamo terminator.

   Napišite funkciju `veclen` u programskom jeziku C. Funkcija treba odrediti duljinu polja zadanog argumentom na način da pobroji koliko elemenata se nalazi prije terminatora. Funkcija treba raditi za polja različitih tipova, a kriterij za detekciju terminatora treba zadati korisnik. Pokažite ispitni program koji vašu funkciju primjenjuje i) na polje cjelobrojnih podataka, te ii) na polje znakova.

2. Implementirajte sljedeće nadogradnje programskog koda iz 6. zadatka vježbe. Ako u problemu prepoznajete neki oblikovni obrazac — navedite ime obrasca te sudionike.

   (a) Napišite metodu razreda `Sheet` koja ispisuje koordinate elementa tablice koji izravno referencira najveći broj drugih elemenata.

   (b) Napišite metodu razreda `Sheet` koja ispisuje koordinate elementa tablice kojeg izravno referencira najveći broj drugih elemenata.

   (c) Napišite komponentu koja na ulazu prima listu elemenata tablice, a na standardnom izlazu ispisuje odgovarajuću razdiobu pod pretpostavkom da svi elementi sadrže nenegativne numeričke vrijednosti. Označimo vrijednosti zadanih elemenata tablice s v_i. Tada vaša komponenta treba ispisati $v_i/(\sum_j v_j + 10^{-5})\ \forall i$. Ispis se treba automatski ponoviti prilikom svake promjene u odabranim poljima tablice.

3. Implementirajte sljedeće nadogradnje programskog koda iz 5. zadatka vježbe.

   (a) Napišite kod za novi izvor brojeva koji generira slučajan niz brojeva iz uniformne distribucije. Granice distribucije te broj generiranih brojeva zadaju se prilikom inicijalizacije izvora.

   (b) Koje obrasce ste koristili u ovom zadatku? Skicirajte dijagram razreda svog rješenja i povežite ga sa sudionicima korištenih obrazaca.

   (c) Na dijagramu razreda dizajnirajte rješenje za dodavanje tri nove akcije: prva akcija u datoteku zapisuje dotada zadnji, druga dotada najveći dok treća zapisuje dotada najmanji element u generiranom nizu. Skicirajte kod za prvu akciju.

Rješenje prvog zadatka

```c
#include <stdio.h>

int termchar(void* p){
  return (*(char*)p)==0;
}

int termint(void* p){
  return (*(int*)p)==0;
}

int veclen(void* p, int sz, int (*is_terminator)(void *)){
  int rv=0;
  while(!is_terminator(p)){
    p = (char*)p + sz;
    ++rv;
  }
  return rv;
}

int main(){
  int vecint[]={12,34,0};
  char vecchar[]="poklon";

  printf("%d␣%d␣\n",
    veclen(vecint, sizeof(int), termint),
    veclen(vecchar, sizeof(char), termchar));
}
```

Rješenje drugog zadatka:

```
class Sheet:
  # ...
  # a)
  def maxrefsout(self):
    cmax_nrefs = None
    rv = None
    for row in range(self.n):
      for col,x in enumerate(self.cells[row]):
        x_nrefs = len(self.getrefs(x))
        if not cmax_nrefs or cmax_nrefs < x_nrefs:
          cmax_nrefs = x_nrefs
          rv = (row,col)
    return rv

  # b)
  def maxrefsin(self):
    cmax_nobs = None
    rv = None
    for row in range(self.n):
      for col,x in enumerate(self.cells[row]):
        x_nobs = len(x.observers)
        if not cmax_nobs or cmax_nobs < x_nobs:
          cmax_nobs = x_nobs
          rv = (row,col)
    return rv
```

```
# c)
# Obrazac: promatrac
# - subjekt: Cell
# - promatrac: Distribution
class Distribution:
  def __init__(self, X):
    self.X = X
    for cell in X:
      cell.attach(self)
    self.update(None)
  def update(self, _):
    mysum = sum(cell.value for cell in self.X) + 1e-5
    print([round(cell.value/mysum,2) for cell in self.X])

# ...

s=Sheet(5,5)

# ...

distrib = Distribution(s.cells[0])
s.set('a0','5')
s.pprint()
print(s.maxrefsin())
print(s.maxrefsout())
```

Rješenje trećeg zadatka:

```
# a)
import random

class Uniformni(Izvor):
  def __init__(self, lower, upper, N):
    self.upper = upper
    self.lower = lower
    self.N = N
    self.N_gen = 0

  def generiraj_broj(self):
    if self.N_gen < self.N:
      self.N_gen+=1
      return random.randint(self.lower, self.upper)
    else:
      return -1

# c)
class ZapisiElement(Akcija):
  def __init__(self, putanja):
    self.putanja = putanja
    self.slijed.dodaj_akciju(self)

  def obavi_akciju(self, slijed):
    element = self.dohvati_element(slijed):
    self.zapisi_element(element)

  def zapisi_element(self.element):
    f = open(self.putanja, "a")
    f.write(element)

class ZapisiZadnji(ZapisiElement):
  def __init__(self, putanja):
    ZapisiElement.__init__(self.slijed, putanja)

  def dohvati_element(self, slijed)
    return slijed[-1]
```