Домаћи

Пре него што почнете прочитајте поглавље [УПУТСВО](#_bookmark0) на крају овог документа. Ту ћете пронаћи упутства како да попуните овај документ и како треба да доставите своја решења задатака из овог домаћег.

# ЗАДАЦИ

1. Прочитајте поглавља 2 и 3 књиге Бјарнеа Строуструпа „*Programming Principles and Practices C++"*. Научите значења свих појмова из тих поглавља одговорите на следећа питања:
   1. Набројте четири дела дефиниције функције:

1. Povratni tip fje – govori koja je povratna vrednost rezultata fje

2. Ime fje

3. Lista parametara – primer (int arg0, int arg1)

4. Telo funkcije – definise akcije koje funkcija izvrsava

* 1. Приликом учитавања са стандардног улаза, шта означава крај једног уноса за целе бројеве и знаковне низове?

Kraj oznacava znak “\n” –newline. Pritiskom na enter automatski se dodaje na kraj unosa.

* 1. Шта је објекат?

Objekat je region memorije sa tipom, koji govori koji tip informacije mozemo cuvati u toj memoriji I koje operacije mozemo izvrsiti nad njim. Imenovani objekat zovemo promenljiva.

* 1. Шта је литерал? Наведите неколико типова литерала.

Literali su podaci sa fiksnim vrednostima ( 1, 2.5, ‘c’ ). String literali su predstavljeni sa “”.

* 1. Шта су променљиве? Наведите основне типове променљивих и њихове уобичајене величине.

Promenljive ili variables su imenovani objekti, imaju specifican tip.

Integer – 4bytes, Character – 1byte, Boolean – 2bytes, Floating Point – 4 bytes, Double Floating Point – 8 bytes, Void – 2bytes, Wide Character – 2 to 4 bytes

* 1. Која је разлика између операција = и ==? Ког типа је њихов резултат?

= je operacija dodele “assignment operator” I koristi se za dodelu vrednosti promenljivoj.

== operator proverava da li su dva data operanda jednaka ili ne, njegova povratna vrednost je true/false

* 1. Шта је дефиниција, а шта декларација?

Definicija dodeljuje memoriju entitetu, snabdeva entitet sa imenom I tipom. Definicija sprovodi sve detalje tog entiteta, ako je fja sta radi, ako je klasa, koja polja I metode ima, ako je promenljiva gde se nalazi.

* 1. Шта је иницијализација и по чему се разликује од доделе?

Inicijalizacija daje promenljivoj inicijalnu vrednost onog momenta kad je kreirana.

Dodela daje vrednost promenljivoj u nekom momentu njenog postojanja.

* 1. Како се знаковни низови надовезују у Це++ језику?

Nadovezivanje se odradjuje pomocu operatora +. To je konkatenacija stringova ( string1 + string 2 = string1string2 )

* 1. Шта је типска безбедност и зашто је важна?

Tipska bezbednost je koriscenje tipova na ispravan nacin, izbegavajuci nesigurne castove i sl. Tipska bezbednost je vazna jer ukoliko se narusi moze doci do overwritovanja informacija (tipa ubacimo int u char I time dodajemo pre toga 3 nekoriscena bajta memorije).

* 1. Да ли је смештање вредности **double** типа у променљиву **int** типа типски безбедно? Зашто?

Nije jer double je 8bajtna vrednost a int 4bajtna, double je vrednost koja cuva decimalne brojeve a int celobrojne vrednosti. Pri smestanju double u int se gubi taj zarez pretvara se u celobrojnu vrednost I zauzima dodatno memorije.

1. Напишите добро структуиран и коментарисан Це++ програм који садржи три функције описане у наставку. Када се овде каже „хексадецимални број“ мисли се на стринг са хексадецималним записом броја.
   1. Функција 1 – прима неозначени цео број и враћа стринг са записом тог броја у хексадецималној форми. Функција треба да баци изузетак у случају неодговарајућег улаза.
   2. Функција 2 – прима два хексадецимална броја (стрингове са хексадецималним записом броја) и враћа њихов збор исто у хексадецималном облику. Функција треба да баци изузетак у случају неодговарајућег улаза.
   3. Функција 3 – прима хексадецимални број и враћа целобројну вредност која одговара том хексадецималном запису. Функција треба да баци изузетак у случају неодговарајућег улаза.
   4. Позовите ове функције из мејн функције. Употребите трај (try) и кеч (catch) механизме.
   5. Проверите исправност вашег програма помоћу примера:
      1. исправног улаза
      2. неисправног улаза
      3. празног улаза
2. Напишите програм који очекује од корисника да унесе две вредности истог основног типа (**short**, **int** или **double**). Те вредности треба сместити у променљиве **val1** и **val2**. Затим треба одредити мањи и већи од та два броја, њихов збир, разлику, производ и количник, и исписати те резултате на стандардни излаз. Употребите три функције, по једну за сваки тип променљивих. Омогућити кориснику да изабере коју од те три функције (та три типа) жели.
3. Напиши програм који рачуна Фибоначијеве бројеве. Формула за генерисање Фибоначијевих бројева дата је у наставку:

Fn = Fn-1 + Fn-2,

где је F1 = 1 и F2 = 2.

На пример, F3 F2 + F1

=

= 2 + 1 = 3.

Приметите да ће у једном тренутку Фибоначијеви бројеви бити превелики за **int** (или било који други) тип и доћи ће до прекорачења. Размислите како да прекинете рачунање бројева када дође до прекорачења, или непосредно пре него што до прекорачења дође, и то имплементирајте.

Напишите три посебне функције за три различита типа: **int**, **long** и **long long**.

**Питања**:

1. Колико првих Фибоначијевих бројева можемо сместити у тип **int**?

Која је вредност највећег?

Najveca vred: 1836311903

Max brojeva 45

1. Колико првих Фибоначијевих бројева можемо сместити у тип

**long**? Која је вредност највећег?

Najveca vred: 1836311903

Max brojeva 45

1. Колико првих Фибоначијевих бројева можемо сместити у тип

**long long**? Која је вредност највећег?

Najveca vred: 7540113804746346429

Max brojeva 91

# УПУТСТВО:

Напишите одговоре у одговарајућа поља. Проширите величину поља ако вам треба више места. Решења за задатке II, III и IV, треба да буде приложено уз овај документ. Све то спакујте у Зип архиву. Архиву треба назвати овако:

***ИМЕ\_ПРЕЗИМЕ\_ИНДЕКС\_domaci.zip***

(Где *ИМЕ, ПРЕЗИМЕ* и *ИНДЕКС* треба заменити са вашим именом, презименом и бројем индекс)

Архива треба да садржи: Датотеке:

**domaci.docx** (са одговорима на питања) Директоријуме:

1. – који садржи пројекат и изворни код за задатак 2
2. – који садржи пројекат и изворни код за задатак 3
3. – који садржи пројекат и изворни код за задатак 4

У решењима задатака треба користити само оне елементе језика Це++ које смо спомињали током прве три недеље на предавањима. Циљ није да се размећете напредним знањем Це++ синтаксе.

**Решење послати у задатом формату предметном асистенту путем емаил-а. У SUBJECT поље уписати: [OOP2 domaci][Име][Презиме][Индекс]. Рок за предају је уторак, 08.11. у поноћ.**

**Решења која нису исправно форматирана или послата на време неће бити прегледана.**

**ВАЖНО**

Пројектни директоријуми (*II*, *III* и *IV*) **НЕ СМЕЈУ** садржати извршне или објектне датотеке!!! Дозвољене су искључиво следеће датотеке:

.vcxproj

.sln

.filters

.cpp

.h

.hpp

**Изоставити** аутоматски направљене директоријуме: *Debug*, *Release*, *ipch* и сл.

6