Raspodijeljene glavne knjige i kriptovalute

# Prva laboratorijska vježba: Bitcoin transakcije

22. listopada 2020.

## Priprema okruženja

Laboratorijsku vježbu možete pisati u bilo kojem programskom jeziku, ali ćete podršku od asistenata imati samo za programski jezik Java. Ako se ipak odlučite za rad u nekom drugom programskom jeziku koji nije kompatabilan s Javom (poput Kotlin-a i Scala-e) morat ćete se sami pobrinuti da pronađete sve potrebne biblioteke te ćete morati napisati ekvivalentne testove kao one dane u početnom kodu.

Preporuča se korištenje razvojnog okruženja *Intellij IDEA*. Kao studenti imate pravo na korištenje edukativne licence za sve *JetBrains* proizvode. Pravo na licencu možete dobiti prijavom putem svoje FER email adrese na stranici `https://www.jetbrains.com/student/`. Uz IDE se preporuča korištenje *Maven* sustava za upravljanje ovisnostima.

Uz svaku laboratorijsku vježbu studenti će dobiti zadatak i početni kod pisan u programskom jeziku Java (verzije 1.8) s Maven podrškom. Unutar projekta će se nalaziti `pom.xml` datoteka u kojoj su zapisane sve ovisnosti projekta. Kako bi postavili projekt potrebno je:

- pokrenuti *Intellij IDEA*,
- kliknuti na "Import Project" te
- navigirati do datoteke `pom.xml` te je otvoriti.

Nakon toga će se projekt otvoriti te će se automatski dohvatiti sve ovisnosti potrebne za normalno pokretanje projekta.

## Uvod

Cilj vježbe je upoznati se s Bitcoin transakcijama i *Bitcoin Script* jezikom. Dobiveni početni kod koristi `bitcoinj` Java biblioteku (čija je dokumentacija dostupna na `https://bitcoinj.github.io`). U razredu `WalletKit` implementirana je logika spajanja na testni blockchain, na kojeg se spaja kroz adresu *bujica.zemris.fer.hr:8080* koristeći `bitcoinj` biblioteku. Kako biste se upoznali s `bitcoinj` bibliotekom, u početnom kodu možete naći primjer implementirane *Pay-to-Public-Key* (P2PK) transakcije unutar razreda `PayToPubKey`. Implementirane su metode za generiranje *locking* (*scriptPubKey*) i *unlocking* (*scriptSig*) skripti - iako se *unlocking* skripta koristi tek pri trošenju novčića u idućoj transakciji. Kako bi se testirale obje skripte, primijetite da se u testovima izvršavaju dvije transakcije - prvo se novčići zaključavaju implementiranom *locking* skriptom, a potom se vraćaju pomoću *unlocking* skripte natrag na vlastitu adresu u sklopu novčanika (pogledajte implementaciju u `ScriptTest#testTransaction` metodi).

## Upute

U sklopu prve laboratorijske vježbe potrebno je izvršiti nekoliko transakcija. Kako bi mogli izvršavati transakcije potrebni su vam novčići. Da bi dohvatili novčiće prvo morate napraviti novčanik koji ima adresu na koju je moguće uplatiti novčić. Novčanik možete napraviti tako da pokrenete `ScriptTests#printAddress` test. Pokretanjem testa `bitcoinj` biblioteka će stvoriti novčanik u direktoriju projekta (samo prvi put kad pokrenete test) pod imenom "wallet". Datoteke `password.spvchain` i `password.wallet` spremaju informaciju o novčaniku i stanje raspodijeljene glavne knjige (*engl. blockchain*). Nakon što se test izvrši pronađite liniju s adresom vašeg novčanika. U trenutku kada vam je poznata adresa vašeg novčanika, novčiće možete dobiti tako da upišete vaš JMBAG i adresu novčanika u formu na adresi https://bujica.zemris.fer.hr/faucet.

## Zadatci

U sklopu ove laboratorijske vježbe potrebno je implementirati *locking* i *unlocking* skripte za zadane transakcije, te pokrenuti odgovarajuće testove implementirane u sklopu početnog koda. Minimalni uvjet za polaganje vježbe je ispravno riješen prvi zadatak.

1. (2 boda) Implementirajte *Pay-to-Public-Key-Hash* (P2PKH) transakciju koja će poslati mali broj novčića na proizvoljnu adresu. Adresu možete generirati upotrebom `bitcoinj` biblioteke tako da stvorite novi `ECKey` objekt. Sav kod morate napisati unutar razreda `PayToPubKeyHash`. Provjerite svoju implementaciju transakcije tako da pokrenete `ScriptTests#testPayToPubKeyHash` test. Izvršenu transakciju možete provjeriti koristeći blockchain explorer na adresi https://bujica.zemris.fer.hr.

2. (1 bod) Napišite transakciju s *locking* skriptom koju mogu otključati dva broja $x$ i $y$ takva da vrijedi.

$$x + y = \text{prve 4 znamenke vašeg JMBAG-a}$$
$$|x - y| = \text{zadnje 4 znamenke vašeg JMBAG-a}$$

   Prilagodite zadnju znamenku vašeg JMBAG-a tako da postoji cijelobrojno rješenje gore navedenog sustava (oba broja moraju biti iste parnosti). Rješenje zadatka se treba nalaziti u `LinearEquationTransaction` razredu. U svojoj implementaciji morate koristiti `OP_ADD` i `OP_SUB`. Provjerite svoju implementaciju koristeći `ScriptTest#testLinearEquation` test.

3. (2 boda) Alice i Bob žele uz Vašu pomoć igrati igru par-nepar koristeći Bitcoin. Pravila igre su sljedeća: sudionici se unaprijed dogovore tko će biti „par" (Alice), a tko „nepar" (Bob); svatko od njih odabere broj: 0 ili 1; ako je zbroj odabranih brojeva paran pobjeđuje „par", inače pobjeđuje „nepar". Da bi spriječili situaciju u kojoj jedan sudionik vidi broj od drugog te sukladno tome promijeni svoj izbor, Alice i Bob slože se da će koristiti shemu obvezivanja (*commitment scheme*):

   1. Odaberi tajni broj $N \in \{0, 1\}$.
   2. Generiraj nonce $R$ veličine $N + 16$ bajtova (dakle 16 ili 17 bajtova).
   3. Izračunaj javno obvezivanje (*public commitment*) $C = \texttt{HASH160}(R)$.

   Na ovaj način osigurali smo da Alice (Bob) ne vidi odabrani broj Boba (Alice) prije početka igre te da ne može naknadno promijeniti svoj odabrani broj. Vaš zadatak je, za zadanu *unlocking* skriptu koja sadrži redom: potpis igrača, $R_A$ i $R_B$, napisati *locking* skriptu koja se uspješno izvršava ako potpis iz *unlocking* skripte pripada pobjedniku igre (Alice ili Bob). *Locking* skripta treba raditi sljedeće:

   1. Osigurati da $R_A$ i $R_B$ odgovaraju redom $C_A$ i $C_B$.
   2. Izračunati $N_A$ i $N_B$ iz redom $R_A$ i $R_B$.
   3. Odrediti tko je pobjednik.
   4. Provjeriti ispravnost potpisa u *unlocking* skripti za javni ključ pobjednika.

## Naputci

- Informacije o naredbama koje Bitcoin Script podržava možete naći na stranici https://en.bitcoin.it/wiki/Script.
- Za potrebe laboratorijske vježbe koristi se privatni testnet na kojem se blokovi rudare s najmanjom težinom (vrijeme potrebno za rudarenje je minimalno). Takav oblik testneta se zove *regtest*, i vrlo je pogodan za testiranje. Regtest je u laboratorijskoj vježbi korišten iz razloga što se rezultati transakcije mogu vidjeti u vrlo kratkom roku (svakih 3 minute se rudari novi blok, za razliku od testneta gdje je brzina rudarenja $\sim 10$ min/blok). Više o regtestu i kako ga možete podesiti na svom računalu možete naći na https://github.com/bitcoinbook/bitcoinbook/blob/develop/ch09.asciidoc i https://samsclass.info/141/proj/pBitc1.htm.

## Predaja

Laboratorijsku vježbu možete rješavati sami ili u grupi od najviše dva studenta. Ako se odlučite raditi laboratorijsku vježbu u grupi, morate poslati zahtjev za grupni rad na adresu rgkk@fer.hr. U zahtjevu morate napisati ime, prezime te JMBAG studenata koji će raditi u grupi. **Prijavu za grupni rad morate poslati do 27.10.2020 u 23:59h.**

Laboratorijsku vježbu predajete putem MS Teamsa tako da učitate, pod „Assignment", *.zip* datoteku koja mora sadržavati samo izvorni kod laboratorijske vježbe. Naziv predane datoteke mora biti u obliku *ime-prezime-jmbag.zip*. Ukoliko radite u grupi, neka naziv datoteke bude ime, prezime i JMBAG jednog od studenata iz grupe. **Rok za predaju laboratorijske vježbe je 4.11.2020 do 23:59h.**

Laboratorijska vježba ocijeniti će se samo na temelju predanog izvornog koda s time da asistenti po potrebi imaju pravo od studenta zahtjevati i usmenu obranu putem MS Teamsa. **Prag za prolaz laboratorijske vježbe je riješen prvi zadatak.**

Pitanja za laboratorijsku vježbu moguća su putem elektroničke pošte rgkk@fer.hr ili konzultacija preko MS Teamsa uz prethodni dogovor putem elektroničke pošte.