

1. Unaprediti interpretator funkcionalnog pseudojezika (čiji je kod u prilogu) dodavanjem mogućnosti prevođenja na LLVM IR. Programski jezik dopušta definisanje funkcija čiji su argumenti, lokalne promenljive i rezultat isključivo celi brojevi. Definicija funkcije počinje ključnom rečju **function** za kojom (opciono) sledi ime rezultatske promenljive i znak **=**, zatim ime funkcije i u zagradama lista argumenata funkcije. Telo funkcije se navodi u okviru vitičastih zagrada i sadrži niz naredbi dodele, izraza i/ili poziva ugrađene funkcije **print** koja služi za ispis. Jezik podržava aritmetičke izraze nad celim brojevima kao i uslovni ternarni operator (**?:**). Od priloženih test primera generisati **main** funkciju u **.ll** datoteci, od koje se prevođenjem **clang**-om zajedno sa pomoćnim C-funkcijama (iz priložene datoteke **function.c**) može dobiti izvršni program.

(a) <code>function main(){ print(100); }</code>	<code>define void @main() { entry: call void @printi(i32 100) ret void }</code>
(b) <code>function main(){ print(1 + 2 * 3); }</code>	<code>define void @main() { entry: call void @printi(i32 7) ret void }</code>
(c) <code>function main(){ print(100); print(1 + 2 * 3); }</code>	<code>define void @main() { entry: call void @printi(i32 100) call void @printi(i32 7) ret void }</code>
(d) <code>function main(){ a = 5; b = 4; print(a + b); print(a - b); print(a * b); print(a / b); }</code>	<code>define void @main() { entry: call void @printi(i32 9) call void @printi(i32 1) call void @printi(i32 20) call void @printi(i32 1) ret void }</code>
(e) <code>function c = proizvod(a, b) { c = a*b; } function main(){ a = 5; b = 6; print(proizvod(a, b)); }</code>	<code>define i32 @proizvod(i32 %a, i32 %b) { entry: %tmpmul = mul i32 %b, %a ret i32 %tmpmul } define void @main() { entry: %calltmp = call i32 @proizvod(i32 5, i32 6) call void @printi(i32 %calltmp) ret void }</code>
(f) <code>function ispisi_kvadrat(n) { a = n*n; print(a); } function main(){ a = 5; ispisi_kvadrat(a); }</code>	<code>define void @ispisi_kvadrat(i32 %n) { entry: %tmpmul = mul i32 %n, %n call void @printi(i32 %tmpmul) ret void } define void @main() { entry: call void @ispisi_kvadrat(i32 5) ret void }</code>
(g) <code>function f = faktorijel(n) { f = n==0 ? 1 : n*faktorijel(n-1); } function main(){ a = 5; print(faktorijel(a)); }</code>	<code>...</code>