Uputstvo za sve grupe

Kreirati CUP specifikaciju koja za prosleđeni ulazni kod kreira apstraktno sintaksno stablo (AST) na osnovu koga se generiše međukod niskog nivoa. Kao osnovu je moguće koristiti CUP specifikaciju kreiranu u 4. laboratorijskoj vežbi (verziju bez korišćenja *error* simbola)

Za implementaciju AST-a koristiti paket AST koji je dat na sajtu predmeta. Paket AST dopuniti dodatnom klasom (ili klasama) potrebnom za konkretnu gramatiku. Za klasu koju sami dodajete u paket AST obavezno najpre definisati u dodatnom tekstualnom fajlu izgled generisanog međukoda niskog nivoa koji treba da generiše funkcija translate. Taj fajl predati zajedno sa projektom.

Grupa 1

Program → **main** *Block* **exit**

Block → *Declarations Expressions*

Declarations → *Declarations Declaration* | *Declaration*

Declaration → Type **ID ;**

Type → **int** | **float** | **bool**

Expressions → *Expressions* **;** *Expression* | *Expression*

Expression → *Assignment* | *ApplyExpression*

Assignment → **ID :=** *ArithmeticExpression*

ArithmeticExpression → *ArithmeticExpression* **+** *TermExpression*
| *ArithmeticExpression* **–** *TermExpression*
| *TermExpression*

TermExpression → **CONST** | **ID**

ApplyExpression → **for ID in [** *NameList* **] apply** *Expression*

NameList → *NameList* , **ID** | **ID**

Napomena: *ApplyExpression* se izvršava za sve promenljive u *NameList*-i. ID u for delu predstavlja privremenu promenljivu u koju se smešta vrednost tekuće promenljive iz *NameList*-e koja se obrađuje.

Grupa 2

Program → **main ( )** *Block*

Block → **{** *Variables Statements* **}**

Variables → *Variables Variable* | *Variable*

Variable → *Type* **ID ;**

Type → **int** | **real** | **boolean**

Statements → *Statements* **;** *Statement* | *Statement*

Statement → *Assignment* | *IfStatement*

IfStatement → **if (** *RelExpression* **) :** *Block ElsePart*

ElsePart → *ElifList* | *ElseStatement* | **ε**

ElifList → *ElifList Elif* | *Elif* | *ElifList ElseStatement*

Elif → **elif (** *RelExpression* **) :** *Block*

ElseStatement → **else :** *Block*

RelExpression → *Term RelOp Term* | *Term*

Term → **ID** | **CONST**

RelOp → **<** | **<=** | **==** | **<>** | **>** | **>=**

Assignment → **ID :=** *Term*

Napomena: Prvi Block se izvršava ukoliko je uslov u **if** delu ispunjen. Ukoliko uslov iz **if**-a nije ispunjen a postoji jedna ili više Elif naredba izvršava se Block iz Elif naredbe čiji je uslov ispunjen. Ukoliko nijedan od prethodnih uslova nije ispunjen izvršava se Block iz Else naredbe ukoliko postoji.

Grupa 3

Program → **program** *Block* **return**

Block → **begin** *VarList Statments* **end**

VarList → *VarList Var* | *Var*

Var → **ID :** *Type* **;**

Type → **integer** | **char** | **string** | **file**

Statements → *Statements* **;** *Statement* | *Statement*

Statement → *Assignment* | *ReadExpression*

Assignment → **ID =** *Expression*

Expression → *ArithmeticExpression* | **open (** *PrimaryExpression* **)**

ArithmeticExpression → *ArithmeticExpression* **+** *PrimaryExpression*
| *ArithmeticExpression* **–** *PrimaryExpression*
| *PrimaryExpression*

PrimaryExpression → **ID** | **CONST**

ReadExpression → **read (ID in ID) do** *Block*

Napomena: U *ReadExpression*-u drugi **ID** predstavlja identifikator fajla. Pretpostaviti da se izvršenjem **open** naredbe sadržaj fajla učita u memoriju (nije potrebno generisati međukod za naredbu **open**) tako da se podaci iz fajla prenose u niz čiji je identifikator taj **ID**. Na poziciji 0 u tom nizu je broj preostalih podataka u nizu, a stvarni podaci iz fajla kreću od pozicije 1. U *ReadExpression*-u treba korišćenjem naredbe **Load_Arr** učitavati elemente niza počevši od pozicije 1.

Grupa 4

Program → **program** *Block* **end**

Block → **{** *Declarations Statements* **}**

Declarations → *Declarations Declaration* | *Declaration*

Declaration → *VariableDeclaration* | *FunctionDeclaration*

VariableDeclaration → **ID :** *Type* **;**

Type → **int** | **float** | **char**

FunctionDeclaration → **ID (** *Parameters* **) =>** *Expression* **;**

Parameters → *Parameters* **,** *Parameter* | *Parameter*

Parameter → **ID :** *Type* | **ID :** *Type* **= CONST**

Statements → *Statements* **;** *Assignment* | *Assignment*

Assignment → **ID :=** *Expression*

Expression → **CONST** | **ID** | *FunctionCall*

FunctionCall → **ID (** *ArgumentsList* **)**

ArgumentsList → *ArgumentsList* **,** *Expression* | *Expression*

Napomena: *FunctionDeclaration* se prevodi u *Expression* zadat tom funkcijom, a početak tog koda je označen labelom koja odgovara nazivu funkcije. *FunctionCall* treba da izvrši skok na tu labelu.

Grupa 5

Experiment → **experiment** *Body* **~experiment**

Body → *Declarations Statements Requirements Execution*

Declarations → VariableDeclaration

VariableDeclaration → *VariableDeclaration* **;** *Variable* | *Variable*

Variable → *Type* **ID ;**

Type → **int** | **double** | **string**

Statements → *Statements* **;** *Statement* | *Statement*

Statement → *Assignment* | *IfStatement*

IfStatement → **if (** *RelExpression* **) : {***Statements***}** ***else {Statements}***

RelExpression → *Term RelOp Term* | *Term*

Term → **ID** | **CONST**

RelOp → **<** | **==**| **>**

Assignment → **ID =** *Expression*

Requirements → **requirements** *NodeNumber* **~requirements**

NodeNumber → **nodes CONST ;**

Execution → **execution** *NodeList* **~execution**

NodeList → *NodeList* ; *NodeDef* | *NodeDef*

NodeDef → **node** *NodeName* **,** *Route* **~node**

NodeName → *name* **ID**

Route → **[** *Waypoints* **]**

Waypoints → *Waypoints* **;** *Waypoint* | *Waypoint*

Waypoint → **WP <** *X* **,** *Y* **,** *Z* **>**

X → *Term*

Y → *Term*

Z → *Term*

Expression → *Expression* **+** *Term* | *Expression* ***** *Term* | *Term*

Napomena: U If naredbi, lista naredbi iz then dela se izvršava ukoliko je uslov različit od 0, a iz else dela u suprotnom. Potrebno je modifikovati ovo pravilo u odnosu na prethodne vežbe, što podrazumeva izmene u leksičkom i sintaksnom analizatoru.

Grupa 6

Model → **model** *Body* **~model**

Body → *Declarations Statements Deployment*

Declarations → *VariableDeclaration*

VariableDeclaration → *VariableDeclaration* **;** *Variable* | *Variable*

Variable → **ID :** *Type* **;**

Type → **int** | **double** | **string**

Statements → *Statements* **;** *Statement* | *Statement*

Statement → *Assignment* | *WhileStatement*

WhileStatement → **while (** *RelExpression* **)** **:** **{***Statements***}** **default { *Statement* }**

RelExpression → *Term RelOp Term* | *Term*

Term → **ID** | **CONST**

RelOp → **less** | **equal** | **greater**

Assignment → **ID :=** *Expression*

Deployment → **deployment** *TaskList ServerList* **~deployment**

ServerList → *ServerList* **;** *ServerDef* | *ServerDef*

ServerDef → **server** *ServerName* **,** *Capacity* **~server**

ServerName → **serverId ID**

Capacity → *Term*

TaskList → *TaskList* **;** *TaskDef* | *TaskDef*

TaskDef → **task** *TaskName* **,** *Demand* **,** *Mapping* **~task**

TaskName → **taskId ID**

Demand → *Term*

Mapping → **executedOn ID**

Expression → *Expression* **+** *Term* | *Expression* ***** *Term* | *Term*

Napomena: Lista naredbi u While petlji se ponavlja sve dok je uslov različit od 0. Kada postane 0, izvršava se samo naredba u default delu. Potrebno je modifikovati ovo pravilo u odnosu na prethodne vežbe, što podrazumeva izmene u leksičkom i sintaksnom analizatoru.

Grupa 7

Diagram → **diagram** *Body* **~diagram**

Body → *Declarations Statements Deployment*

Declarations → *VariableDeclaration*

VariableDeclaration → *VariableDeclaration* **;** *Variable* | *Variable*

Variable → *Type* **ID ;**

Type → **int** | **double** | **string** | **bool**

Statements → *Statements* **;** *Statement* | *Statement*

Statement → *Assignment* | *DoStatement*

DoStatement → ***do*** **(** *Statements* **) while (** *RelExpression* **) times (CONST)**

RelExpression → *Term RelOp Term* | *Term*

Term → **ID** | **CONST**

RelOp → **<** | **==** | **>**

Assignment → **ID :=** *Expression*

Deployment → **deployment** *ServiceList ServerList* **~deployment**

ServerList → *ServerList* **;** *ServerDef* | *ServerDef*

ServerDef → **server** *ServerName* **,** *Instances* **,** *Environment* **~server**

ServerName → **serverName ID**

Instances → **numInstances** *Term*

ServiceList → *ServiceList* **;** *ServiceDef* | *ServiceDef*

ServiceDef → **service** *ServiceName Environment Allocation* **~service**

ServiceName → **serviceName ID**

Environment → **cloud** | **edge**

Allocation → **executedBy ID**

Expression → *Expression* **+** *Term* | *Expression* ***** *Term* | *Term*

Napomena: Lista naredbi u Do-While petlji se izvršava ukoliko je uslov true, a izvršenje ponavlja maksimalno puta koliko je vrednost celobrojne konstante u times delu. Potrebno je modifikovati ovo pravilo u odnosu na prethodne vežbe, što podrazumeva izmene u leksičkom i sintaksnom analizatoru.

Grupa 8

Strategy → **strategy** *Body* **~strategy**

Body → *Declarations Statements ServiceList*

Declarations → **declaration** *VariableDeclaration* **~declaration**

VariableDeclaration → *VariableDeclaration* **;** *Variable* | *Variable*

Variable → *Type* **ID ;**

Type → **int** | **double** | **string** | **bool** | **char**

Statements →*Statements* **;** *Statement* | *Statement*

Statement → *Assignment* | *WhileStatement*

WhileStatement → **repeat (** *Term* **)** **{** *Statements* **}**

RelExpression → *Term RelOp Term* | *Term*

Term → **ID** | **CONST**

RelOp → **less** | **equal** | **greater**

Assignment → **ID =** *Expression*

Instances → **numInstances** *Term*

ServiceList → *ServiceList* **;** *ServiceDef* | *ServiceDef*

ServiceDef → **service** *ServiceName Instances Allocation AdaptationRule* **~service**

ServiceName → **serviceName ID**

Allocation → **executedBy ID**

AdaptationRule → **if** *Condition* **then** *Response*

Condition → *RelExpression*

Response → **scale** Term | **redeployOn ID** | **optimize**

Expression → *Expression* **+** *Term* | *Expression* ***** *Term* | *Term*

Napomena: Izraz u uslovu Repeat petlje može biti integer (**modifikacija u odnosu na prethodnu vežbu**). Kolika je vrednost uslova, toliko puta se lista naredbi za Repeat ponavlja. Potrebno je modifikovati ovo pravilo u odnosu na prethodne vežbe, što podrazumeva izmene u leksičkom, sintaksnom i semantičkom analizatoru.

Grupa 9

Program → **main ( )** *Block*

Block → **{** *Declarations StatementList* **}**

Declarations → *Declarations VarDecl* | **ε**

VarDecl → *Type NameList* **;**

NameList → **ID** | *NameList* **, ID**

Type → **int** | **char** | **float**

StatementList → *StatementList Statement* | *Statement*

Statement → *CaseStatement* | **ID** = *Expression* **;** | *Block*

CaseStatement → **case (** *Expression* **) {** *WhenStatementList* **}**

WhenStatementList → *WhenStatementList WhenStatement* | *WhenStatement*

WhenStatement → **when CONST :** *Statement*

Expression → *Expression AddOperator Term* | *Term*

AddOperator → **+** | **-**

Term → **ID** | **CONST** | **(** *Expression* **)**

Napomena: U okviru *CaseStatement* upravljačke strukture izvršava se naredba iz prvog *WhenStatement* dela gde je konstanta **CONST** jednaka rezultatu izraza *Expression* u uslovu *CaseStatement* upravljačke strukture. Posle izvršenja te naredbe izlazi se iz *CaseStatement*-a (tj. izvršava se najviše jedan *WhenStatement* deo).

Grupa 10

Program → **program** *Block* **.**

Block → **begin** *Variables StatementList* **end**

Variables → *Variables Declaration* | **ε**

Declaration → *NameList* **:** *Type* **;**

NameList → *NameList* **,** **ID** | **ID**

Type → **integer** | **char** | **real** | **boolean**

StatementList → *Statement* | *StatementList Statement*

Statement → *WhileLoop* | **ID :=** *Expression* **;** | *Block*

WhileLoop → **while** *Expression* **:** *Statement* **else** *Statement*

Expression → *Expression* **or** *AndExpression* | *AndExpression*

AndExpression → *AndExpression* **and** *RelExpression* | *RelExpression*

RelExpression → *Term RelOp Term* | *Term*

RelOp → **<=** | **==** | **>=**

Term → **ID** | **CONST** | **(** *Expression* **)**

Napomena: Prva naredba u *While* petlji se izvršava sve dok je ispunjen uslov petlje (*Expression*). Kada taj uslov više nije ispunjen izvršava se tačno jednom naredba u *else* delu petlje i izlazi se iz petlje.

Grupa 11

Program → **main ( )** *Block*

Block → **{** *Declarations StatementList* **}**

Declarations → *Declarations VarDecl* | **ε**

VarDecl → *Type NameList* **;**

NameList → **ID** | *NameList* **, ID**

Type → **int** | **char** | **float** | **bool**

StatementList → *StatementList Statement* | *Statement*

Statement → *RedoLoop* | **ID** = *Expression* **;** | *Block*

RedoLoop → **loop (** *Expression* **) {** *Statement* **redo (** *Expression* **) ;** *Statement* **}**

Expression → *Expression* **||** *AndExpression* | *AndExpression*

AndExpression → *AndExpression* **&&** *RelExpression* | *RelExpression*

RelExpression → *Term RelOp Term* | *Term*

RelOp → **<=** | **==** | **>=**

Term → **ID** | **CONST** | **(** *Expression* **)**

Napomena: Spoljašnja petlja (*loop*) u okviru *RedoLoop* upravljačke strukture se izvršava sve dok je ispunjen prvi uslov *Expression*. Naredba (*Statement*) pre ključne reči **redo** se izvršava sve dok je ispunjen uslov *Expression* posle ključne reči **redo**. Kada taj uslov više nije ispunjen izvršava se druga naredba (*Statement*).

Grupa 12

Program → **program** *Block* **.**

Block → **begin** *Variables StatementList* **end**

Variables → *Variables Declaration* | **ε**

Declaration → *NameList* **:** *Type* **;**

NameList → *NameList* **,** **ID** | **ID**

Type → **integer** | **char** | **real** | **boolean**

StatementList → *Statement* | *StatementList Statement*

Statement → *SelectStatement* | **ID :=** *Expression* **;** | *Block*

SelectStatement → **select begin** *CaseList* **end**

CaseList → *CaseList Case* | *Case*

Case → **case** *Expression* **=>** *Statement*

Expression → *Expression* **or** *AndExpression* | *AndExpression*

AndExpression → *AndExpression* **and** *RelExpression* | *RelExpression*

RelExpression → *Term RelOp Term* | *Term*

RelOp → **<** | **==** | **>**

Term → **ID** | **CONST** | **(** *Expression* **)**

Napomena: U okviru *SelectStatement* upravljačke strukture izvršava se naredba iz svakog *Case* dela čiji je uslov *Expression* ispunjen.