KHULNA UNIVERSITY OF ENGINEERING & TECHNOLOGY
B.Sc. Engineering 3rd Year 2nd Term Examination, 2016
Department of Computer Science and Engineering
CSE 3201
Operating Systems

TIME: 3 hours FULL MARKS: 210

N.B. i) Answer **ANY THREE** questions from each section in separate scripts.
ii) Figures in the right margin indicate full marks.

SECTION A

(Answer **ANY THREE** questions from this section in Script A).

1. a) Define deadlock. Using multiple 'Resource-allocation graphs', discuss the scenarios of deadlock. Also explain the conditions that may cause deadlock. (15)
   b) Explain Banker's algorithm with an appropriate example. Also discuss preconditions and data structures of it. (13)
   c) Discuss the issues of resource preemption to eliminate deadlock. (07)

2. a) Make a comparison between physical and logical addresses. How instructions and data can be combined to memory address? Explain with necessary diagram. (13)
   b) Describe the following storage allocation algorithms: (07)
   (i) First fit (ii) Best fit (iii) Worst fit
   c) How does paging ensure non-contiguous memory allocation? Discuss it precisely. How does TLB facilitate to implement paging hardware? Explain with diagram. (15)

3. a) Define distributed system. Discuss the merits of it. (07)
   b) What is distributed file system (DFS)? Discuss about DFS structure. (08)
   c) Define 'stateful file service' and 'stateless file service'. Make a comparison between stateful and stateless service. (10)
   d) What are the techniques to avoid repeated collisions over a communication network? Explain them. (10)

4. a) Define access matrix. How can it be implemented? (08)
   b) How does RSA cryptosystem ensure secure communication over insecure medium? Explain with example. (10)
   c) What is meant by user authentication? How does digital signature technique ensure user authentication? Explain. (07)
   d) What is 'man-in-the-middle' attack? Discuss how a boo-sector computer virus affects the operating system. (10)

SECTION B

(Answer **ANY THREE** questions from this section in Script B)

5. a) What is Microkernel? Draw a block diagram of Microkernel System Structure. (06)
   b) What do you mean by Medium term scheduling? Explain with diagram. (09)
   c) What is context switching? Explain with Process Control Block. (07)
   d) Write down the pseudo code for creating child process from parent. How do you make this child zombie? (08)
   e) What are the benefits of multithreading? (05)

6. a) "Serial portion of an application has disproportionate effect on performance gained by adding additional cores" – explain this statement using Amdahl's law. (08)
   b) What is Round Robin CPU scheduling? Consider the following processes arrive at nearly same time and arrive sequentially. (14)

Processes	Burst Time
P1	6
P2	3
P3	1
P4	7
P5	2

Now, calculate average turn around and waiting time for time quantum 3 and 5. Which time quantum is the most suitable for this system?

c) What is multilevel feedback queue? Explain multilevel feedback queue with the following scenario: (13)

Level 1 queue Q1: RR scheduling with 2 milliseconds of time quantum.
Level 2 queue Q2: RR scheduling with 4 milliseconds of time quantum.
Level 3 queue Q3: RR scheduling with 8 milliseconds of time quantum.
Level 4 queue Q4: SJF scheduling.

Suppose, 4 processes arrive at nearly same time and they arrive sequentially. Their names and burst time are as follows:

Processes	Burst Time
A	3
B	8
C	15
D	3

7. a) Define the following terms: (09)
   i) Soft real time systems.
   ii) Hard real time systems.
   iii) Exponential averaging.

   b) State the four conditions to hold for a good solution with mutual exclusion. (05)

   c) What is semaphore? Solve the producer-consumer problem with semaphore. (15)

   d) In Dining Philosopher problem, what will happen if all the five philosophers take their left fork simultaneously? (06)

8. a) What are the drawbacks of linked list allocation scheme of file? How can these be solved by FAT? Explain with simple example. (10)

   b) What is i-node? What are the benefits of i-nodes over FAT? (05)

   c) What is demand paging? Suppose, memory access time = 200 nanoseconds, average page-fault service time = 8 milliseconds and there is one page fault in every 4,00000 memory access. Find out Effective Access Time. (12)

   d) Apply LRU page replacement algorithm to the following reference string: (08)

   7 7 0 0 1 2 3 7 1 2 4 6 5 1 1 2

   There are 3 page frames. Show the number of page faults.

KHULNA UNIVERSITY OF ENGINEERING & TECHNOLOGY
B.Sc. Engineering 3rd Year 2nd Term Examination, 2016
Department of Computer Science and Engineering
CSE 3211
Compiler Design

TIME: 3 hours FULL MARKS: 210

N.B. i) Answer **ANY THREE** questions from each section in separate scripts.
ii) Figures in the right margin indicate full marks.

SECTION A

(Answer **ANY THREE** questions from this section in Script A)

1. a) What are the phases of a compiler? Why do you divide the compiler design into different phases? Show the contents of different phases for the following statement: (15)

$$value = Base + rate * 60;$$

b) Define ambiguous grammar. Show that following grammar is ambiguous. (06)

$$S \to S(S)S \mid \in$$

c) What is token and lexeme? Write a program in flex to count the number of statements, lines and identifiers in a source program. (10)

d) Draw a transition diagram for an unsigned number. (04)

2. a) Define 'dangling-else' problem with example. (07)

b) Define left factoring. Why do you need to eliminate left factoring? Eliminate left factoring from the following grammar: (07)

$$A \to abB \mid aB \mid cdg \mid cdeB \mid cdfB$$

c) Consider the following grammar of an arbitrary programming language. (capital letters are non terminals and small letters are terminals) (15)

$$A \to pBqArA \mid sBtA \mid uCv$$
$$C \to AwC \mid \in$$
$$B \to m$$

i) Calculate FIRST and FOLLOW set.
ii) Develop LL(1) parse table.
iii) Show the Stack movement for the input string "smtuuvwv".

d) "A left recursive grammar cannot be LL(1) grammar"–justify the statement. (06)

3. a) What can be the contents of the stack for LL(1) parser? What are the actions taken by the parser if the top of stack is a non terminal X ? (10)

b) Define augmented grammar and closure of items with example. (07)

c) What are the contents of goto and action table for a shift reduce parser? How can you calculate them? (08)

d) Define activation tree and activation record. Draw the activation tree for the following program segment. (10)

```
factorial(int n){
  if(n == 1) return 1;
  else return (n*factorial(n-1));
}
main(){
  int x = factorial(4);
}
```

4. a) Define intermediate code. Define the semantic rule for type conversion for the following production rule: (10)

$$E \to E1 + E2$$

b) Let A be a 3 dimensional array of size $10 \times 20 \times 30$ where Low1 = Low2 = Low3 = 0 and base =100. Develop the three address code for $x = A[i][j][k]$. (10)

c) Consider the following code segment: (15)

```
A = B*C+D; sum = 0;
while A do
  begin
    A = A-D;
    Sum = sum+A;
  end;
```

i) Generate the three address code.
ii) Implement the code using quadruples and triples.
iii) Develop the semantic rule for while statement.

SECTION B

(Answer **ANY THREE** questions from this section in Script B)

5. a) Define lookahead symbol. What are the components of a context free grammar (CFG)? (07)
   b) Consider an arithmetic expression (represented in Infix notation) contains the operators: +, –, *, and /. Now construct an unambiguous grammar to evaluate the expression. (08)
   c) Define predictive parsing. Consider the following grammar: (13)

$$
\begin{aligned}
\textit{type} &\rightarrow \textit{simple} \\
&\mid \uparrow\textbf{id} \\
&\mid \textbf{array}\ [\textit{simple}]\ \textbf{of}\ \textit{type} \\
\textit{simple} &\rightarrow \textbf{integer} \\
&\mid \textbf{char} \\
&\mid \textbf{num dotdot num}
\end{aligned}
$$

   Write down the pseudo-code for the predictive parser that validates an input string which follows the syntax of the above grammar.
   d) Construct Directed Acyclic Graph (DAG) for the following statement: (07)

$$a + a * (b - c) + (b - c) * d + d * (e + f)$$

6. a) What is type system? When do we need dynamic checking? Explain with example. (08)
   b) Suppose a desk calculator reads an input line containing an arithmetic expression involving digits, parentheses, the operators '+' and '*' followed by a dollar ($) sign and prints the value of the expression. (10)
      i) Write down the syntax directed definition for the calculator.
      ii) Draw the annotated parse tree for the input $3 * 5 + 4 * 9\$$.
   c) Write type expressions for array of pointers to integers ranging from 1 to 100. (07)
   d) Suppose a language consists of a sequence of declarations followed by a single expression. Types of declaration are integer and char. Write down the translation scheme for checking the type of declarations. (10)

7. a) "The nature of the instruction set of the target machine may determine the efficiently of the target code" – justify the statement. (08)
   b) Suppose a particular machine requires register-pairs (an even and next odd numbered register) for integer multiplication and division. The even register contains the value of the operand (operand denotes multiplicand/dividend). The instruction has the following format: (08)

   OP Source, Destination.

   Now consider an arithmetic expression:

$$a = b + c * d - e / f$$

   Generate the optimal machine-code sequences for the above expression.
   c) What is a basic block? How are the basic blocks determined within a program? (07)
   d) What is the role of register descriptor and address descriptor in code generation algorithm? Show the generated code along with the contents of register descriptor and address descriptor for the following code segment for a simple machine model. (12)

$$
\begin{aligned}
T &= A - B \\
U &= A - C \\
V &= T + U \\
W &= V + U
\end{aligned}
$$

8. a) Explain the following code-improving transformation: (06)
   (i) Constant folding, (ii) Algebraic simplification.

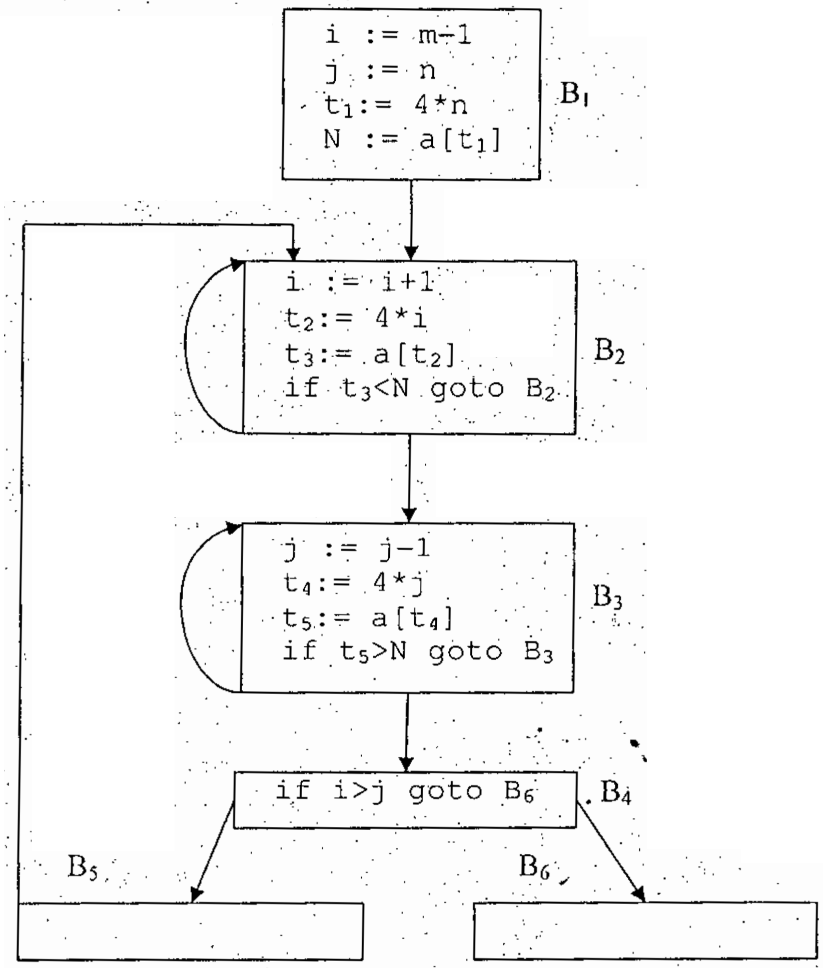
b) Explain 'Structure Preserving Transformation' with proper example. (06)

c) Consider the following code segment: (13)

```
/*code for s*/         /*code for p*/         /*code for q*/
action 1               action 3               action 4
call q                 call q                 return
action 2               return
halt
```

The code for these procedures starts at address 100, 200, and 300 respectively. The stack starts at 600 and each action instruction takes 10 bytes. Show the stack allocation when the target code is produced.

d) Apply the following techniques on the graph given below to improve the code: (10)
   i) induction variables elimination
   ii) reduction in strength.



i := m-1
j := n
t_1 := 4*n
N := a[t_1]
B_1
i := i+1
t_2 := 4*i
t_3 := a[t_2]
if t_3<N goto B_2
B_2
j := j-1
t_4 := 4*j
t_5 := a[t_4]
if t_5>N goto B_3
B_3
if i>j goto B_6
B_4
B_5
B_6