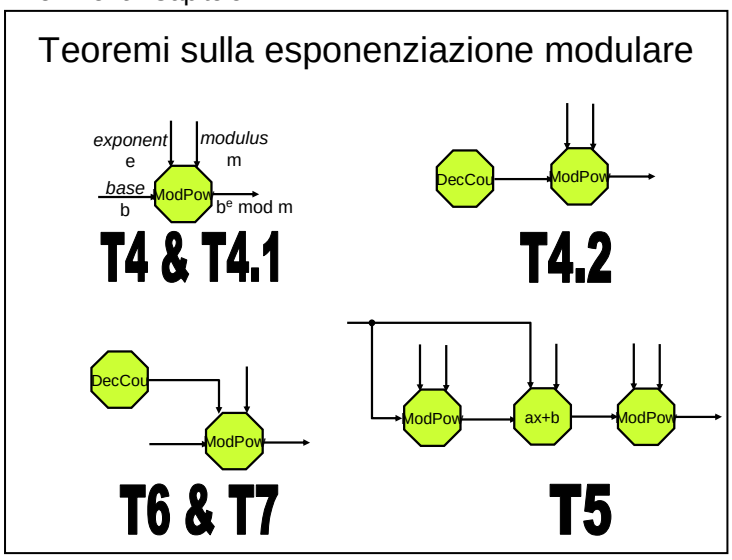
4. Esercitazioni consigliate

4.1 Teoremi sull'esponenziazione modulo un numero primo

Obiettivo formativo – Verificare le proprietà dell'esponenziazione modulo un numero primo
Riferimenti: Capitolo 4

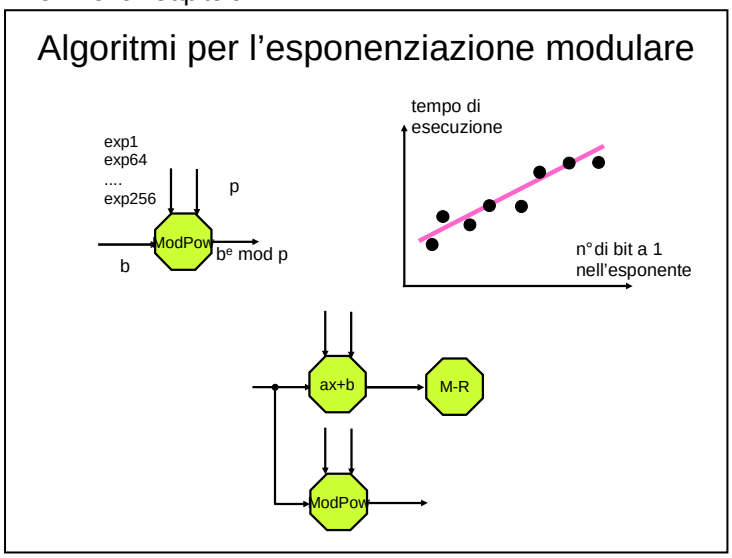


Esperimenti:

1. Prendere atto che il componente ModPow esegue esponenziazioni modulari: i valori della base, dell'esponente e del modulo possono essere o forniti in ingresso o digitati/incollati nella riga omonima della scheda di In/Out.
2. Verificare T4 imponendo m=17, e=16, b scelto a caso minore di 17. Ripetere la prova con un diverso valore di b. Verificare T4.1 imponendo m=17, b scelto a caso minore di 17, e=x e poi e=x+16, con x scelto a caso minore di 17.
3. Collegare un DecCounter (questo blocco fornisce il risultato del conteggio come BigDecimal) all'ingresso b di ModPow, Inizializzare a 0 il Counter e configurare ModPow con e=8, m=17; eseguire lo schema sedici volte per verificare T4.2. Controllando i risultati sul log del mentore, individuare quali basi sono non-residui quadratici e quali sono residui quadratici modulo 17.
4. Prendere atto che il file di log può essere azzerato con il pulsante "clear" della vista del mentore. Scegliere a caso un non-residuo quadratico e verificare con sedici esperimenti che è anche una radice primitiva di 17 (T6, T7). Giustificare il risultato calcolando a mano $\Phi(p-1)$.
5. Prendere atto con alcune prove che il componente "ax+b" accetta in ingresso tre BigDecimal e restituisce in uscita un BigDecimal (quest'ultimo, se necessario, verrà automaticamente trasformato in BigInteger dal componente a valle). Verificare T5 istanziando lo schema indicato in figura e configurandone opportunamente i componenti con numeri appartenenti a Z^*_{17}.

4.2 Algoritmi per l'esponenziazione modulare

Obiettivo formativo – Verificare alcune proprietà notevoli degli algoritmi per l'esponenziazione modulare. Verificare le proprietà di un *safe prime*.
Riferimenti: Capitolo 4

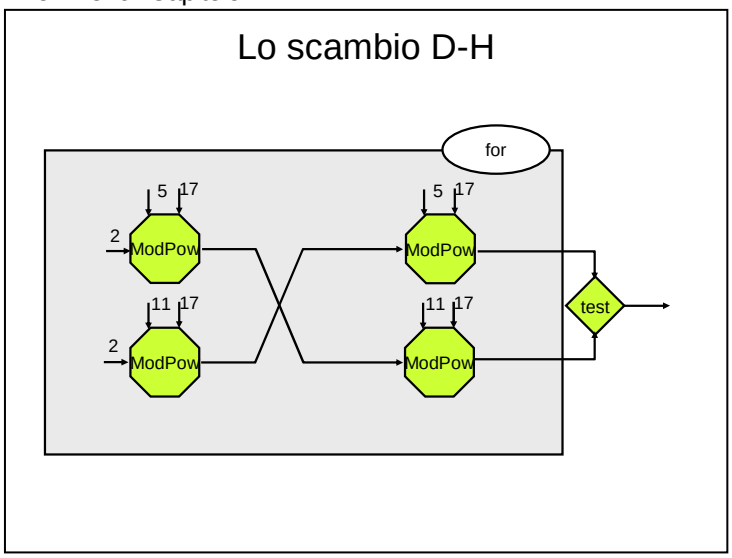


Esperimenti:

1. Prelevare il file pge.txt dalla pagina S-vLab del sito Lia e caricarlo nella cartella file del progetto. Aprire il file, copiare il BigInteger denominato "p" ed incollarlo nel campo *modulus* della scheda In/Out di un ModPow. Copiare il BigInteger "g" ed incollarlo nel campo *base*. Copiare il BigInteger "exp1" ed incollarlo nel campo *exponent*. Fare un certo numero di prove ed annotare l'ultimo valore del tempo di esecuzione. Ripetere con "exp64", "exp128", "exp192", "exp256"; graficare i risultati ottenuti e giustificarli.
2. Il "p" impiegato in precedenza è un safe prime. Verificarlo sia individuando il corrispondente primo di Sophie Germain (il blocco denominato "ax+b" consente di calcolare 0,5 p – 0,5) , sia controllando che è congruente a 5 mod 6.
3. Ripetere la prova con p=17.

4.3 Scambio di Diffie-Hellman

Obiettivo formativo – Imparare a programmare in Java l'accordo su una chiave segreta tramite lo scambio D-H.
Riferimenti: Capitolo 4

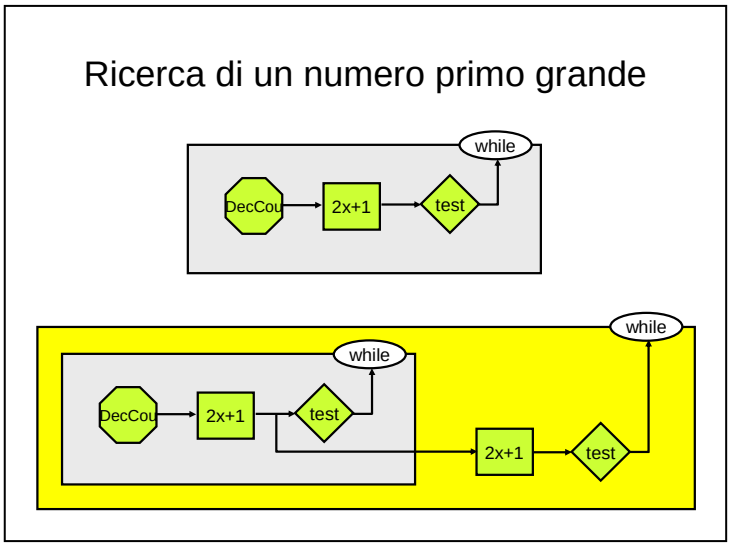


Esperimenti:

1. Istanziare una BOX-for e configurarla per l'esecuzione di un solo ciclo. Collocare all'interno della BOX quattro ModPow, connetterli in modo da realizzare uno scambio D-H e configurarli con i dati indicati in figura.
2. Introdurre un blocco test tra le due uscite, eseguire la BOX e verificare l'eguaglianza dei dati calcolati dai due corrispondenti.
3. Esaminare la scheda codice della BOX.
4. Fare click sulla BOX con il pulsante destro e selezionare il comando "Export Java Code". Prendere atto che il codice della BOX viene salvato nella cartella Java Sample Code.
5. Copiare il file e trasportarlo all'interno di un project dell'ambiente di sviluppo di Eclipse. Modificare il codice esemplificativo, inserendolo all'interno di una definizione di classe (con lo stesso nome del file) e di un metodo *main*. Prendere atto degli errori segnalati automaticamente dall'IDE e correggerli, accettando i suggerimenti relativi all'importazione dei package necessari ed alla ridenominazione delle variabili duplicate. Aggiungere, in coda alle altre istruzioni, una stampa a video dei risultati che si vogliono confrontare e mettere in esecuzione il programma.

4.4 Generazione di numeri primi

Obiettivo formativo – Prendere atto della distribuzione dei numeri primi. Valutare la probabilità che un numero scelto a caso sia primo. Sperimentare un metodo per la generazione di un safe prime.
Riferimenti: Capitolo 4



Esperimenti:

1. Sperimentare il procedimento per la generazione di un numero primo grande, istanziando all'interno di una BOX-while un componente DecCounter (inizializzato con il valore di 2^28, che può essere calcolato separatamente con un ModPow), un componente "ax+b" (configurato con a=2 e b=1, così da produrre in uscita un numero sicuramente dispari), e un test di Miller-Rabin.
2. Eseguire la BOX più volte, al fine di generare più numeri primi. Individuare, grazie al log prodotto dal mentore, il numero primo ottenuto al termine di ogni ciclo "while". Verificare che i primi così trovati fanno parte della lista dei numeri primi che può essere scaricata dal sito "The Prime Pages", accessibile attraverso l'Esperto.
3. Analizzare la scheda Codice del test di Miller-Rabin. Accedere a Javadoc e prendere atto dei parametri del metodo.
4. Istanziare una BOX-while e collocare al suo interno una seconda BOX-while che contenga un DecCounter, un componente "ax+b" ed il test di Miller-Rabin, collegati e configurati come descritto al punto 1. Collegare l'uscita della box più interna ad un secondo componente "ax+b", configurato con a=2 e b=1, ed un test di Miller-Rabin, così che la box esterna termini la sua esecuzione dopo aver trovato un safe prime.
5. Verificare che il safe prime generato al punto 4 è congruente a 5 modulo 6.

REPORT N.4

4.1 Teoremi sull'esponenziazione modulo un numero primo

Immagine dello schema studiato	**Osservazioni:**

4.2 Algoritmi per l'esponenziazione modulare

Immagine dello schema studiato	**Osservazioni:**

4.3 Scambio di Diffie-Hellman

Immagine dello schema studiato	**Osservazioni:**

4.4 Generazione di numeri primi

Immagine dello schema studiato	**Osservazioni:**