

# SISTEM PEMODELAN SIMULASI PERMOHONAN KREDIT MENGGUNAKAN ALGORITMA BANKER

Utawi Handika Sari<sup>1</sup>

Linda Wahyuni<sup>2</sup>

*e-mail* : [ika@potensi-utama.ac.id](mailto:ika@potensi-utama.ac.id) , [linda@potensi-utama.ac.id](mailto:linda@potensi-utama.ac.id)

Diterima : 14 Desember 2010 / Disetujui : 21 Desember 2010

## ABSTRACT

*Deadlock adalah proses menunggu pada satu kejadian tertentu yang tak akan pernah terjadi. Sekumpulan proses berkondisi deadlock bila setiap proses yang ada dikumpulkan itu menunggu suatu kejadian yang hanya dapat dilakukan proses lain yang juga berada di kumpulan itu. Algoritma banker mencegah terjadinya deadlock yang memutuskan apakah menyetujui atau menunda permohonan kredit oleh customer (nasabah). Ketika customer (nasabah) memohon kredit, maka permohonan kredit tersebut harus diperiksa oleh banker. Cara kerja Algoritma banker ada 2 cara yaitu Algoritma safety dan Algoritma resource Request. Algoritma safety adalah cara yang dipakai oleh banker untuk memeriksa keadaan system setelah peminjaman, apakah berada dalam kondisi safe atau tidak. Sedangkan Algoritma Resource Request adalah cara yang dipakai peminjaman sumber daya disetujui atau ditunda.*

*Kata Kunci* : Algoritma, Sistem, Deadlock, Simulasi

## PENDAHULUAN

Selama beberapa dekade yang lalu simulasi mengalami perkembangan yang cukup pesat seiring dengan perkembangan dan menggunakan komputer dan program-programnya. Simulasi ini terutama digunakan untuk menguraikan dan menyelesaikan persoalan-persoalan yang timbul dalam kehidupan nyata, baik disektor publik maupun dalam berbagai sektor bisnis.

- 
1. **Dosen Jurusan Komputerisasi Akuntansi, STMIK Potensi Utama.**  
Jl. K.L. Yos Sudarso Km. 6,5 No. 3 A Tj. Mulia Medan.
  2. **Dosen Jurusan Komputerisasi Akuntansi, STMIK Potensi Utama.**  
Jl. K.L. Yos Sudarso Km. 6,5 No. 3 A Tj. Mulia Medan.

Suatu proses disebut *deadlock* jika proses menunggu satu kejadian tertentu yang tidak akan pernah terjadi. Sekumpulan proses berkondisi *deadlock* bila setiap proses yang ada di kumpulan itu menunggu suatu kejadian yang hanya dapat dilakukan proses lain yang juga berada di kumpulan itu. *Deadlock* terjadi ketika proses-proses mengakses secara eksklusif sumber daya, sedang menggenggam sumber daya dan meminta sumber daya lain (yang sedang dipegang oleh proses lain). Salah satu metode untuk mencegah *deadlock* adalah algoritma Banker.

Algoritma Banker dikemukakan oleh Edsger W.Dijkstra dan merupakan salah satu metode untuk menghindari *deadlock*. Algoritma ini disebut algoritma Banker karena memodelkan sebuah bank di kota kecil yang berurusan dengan sekumpulan nasabah yang memohon kredit. Analogi dari algoritma Banker dengan sistem operasi adalah, nasabah merupakan proses-proses yang sedang berjalan, uang (dana yang dimiliki bank) merupakan sumber daya, dan bankir merupakan sistem operasi. Setiap nasabah memiliki batas kredit. Apabila seorang nasabah telah mencapai batas kredit pinjaman, maka diasumsikan nasabah tersebut telah menyelesaikan semua permasalahan bisnisnya dan dapat mengembalikan semua pinjamannya kepada bank.

Setiap nasabah dapat memohon kredit pada suatu waktu dan bankir dapat menyetujui atau menolak permohonan tersebut. Jika ditolak, nasabah masih menggenggam dana yang telah dipinjamkan untuknya dan menunggu beberapa waktu sampai permohonannya dapat disetujui. Bankir hanya memberikan permintaan yang menghasilkan *state* selamat. Permohonan kredit yang akan menghasilkan *state* tidak selamat secara berulang ditolak sampai permohonan tersebut dapat dipenuhi. Persetujuan atau penolakan permohonan kredit ditentukan dengan menggunakan algoritma *Safety* dan algoritma *Resource Request*.

## PERMASALAHAN

### 1. Identifikasi Masalah

Adapun identifikasi masalah yang terdapat dalam penelitian ini yaitu :

1. Kurangnya pelayanan terhadap nasabah yang menginginkan kredit.
2. Keterlambatan pelayanan dalam proses permohonan kredit untuk nasabah.

### 2. Perumusan Masalah

Uraian perumusan masalah dari penelitian ini adalah:

1. Bagaimana menerapkan algoritma Banker yang terdiri dari algoritma *Safety* dan algoritma *Resource Request* untuk menentukan keputusan apakah sumber daya dipinjamkan atau tidak.

2. Bagaimana membuat atau menggambar objek-objek simulasi dan merancang animasi pergerakan objek dalam sebuah perangkat lunak simulasi.

### 3. Batasan Masalah

Adapun batasan-batasan masalah yang diberikan agar penyelesaian masalah tidak terlalu kompleks adalah sebagai berikut:

1. Data *Input* dari perangkat lunak adalah,
  - a. Jumlah *resource* (sumber dana) ada 5 jenis nama mata uang.
  - b. Setiap *resource* jenis mata uang, Banker hanya tersedia nilai pinjam 100 dana
  - c. Batas pinjam bagi nasabah hanya 1 sampai 30 dana
2. Kondisi simulasi (kondisi awal, permohonan *resources*, besarnya *resources* dan pengembalian *resources*) dapat di-*input* oleh *user* atau dihasilkan secara acak (*random*) oleh komputer.
3. Jumlah tempat pelayanan nasabah (*counter*) dibatasi sebanyak 5 buah.
4. Perangkat lunak akan menampilkan laporan mengenai data-data nasabah yang melakukan transaksi.
5. Laporan tersebut dapat disimpan ke dalam bentuk *text file*.
6. Kecepatan animasi dapat diatur secara manual.
7. Waktu atau lamanya proses simulasi dapat di-*input* oleh *user* dan dibatasi maksimal 3 digit.
8. Menggunakan aplikasi bahasa pemrograman Microsoft Visual Basic 6.0.

## ANALISA DAN PERANCANGAN

### 1. Analisa Sistem Yang Berjalan

Adapun analisa dari simulasi algoritma Banker mencakup alur kerja perangkat lunak, pembuatan objek gambar simulasi dan proses kerja algoritma Banker yang terdiri atas algoritma *Safety* dan algoritma *Resource Request*.

#### 1.1. Input

Analisa input yang digunakan untuk menganalisis pemodelan simulasi ini adalah :

1. Banyaknya *resource* yang tersedia oleh banker.
2. Batas *Maksimum* dan besar alokasi untuk setiap *customer*.

## 1.2. Proses

Untuk analisis proses dilakukan dengan menggunakan algoritma Banker (algoritma *Resource Request* dan algoritma *Safety*) dimana untuk memutuskan apakah permohonan kredit disetujui (*approved*) atau permohonan kredit ditunda (*pending*).

## 1.3. Output

Hasil dari analisis output berupa catatan atau hasil perhitungan menggunakan algoritma *Resource Request* dan algoritma *Safety*, dan melihat *customer* mana yang disetujui (*approved*) atau yang ditunda (*pending*).

## 2. Evaluasi Sistem Yang Berjalan

Dari hasil analisa sistem yang berjalan masih terdapat kelemahan pada perhitungan menggunakan algoritma *Resource Request* dan algoritma *Safety*. Dimana hasil yang didapat masih dapat terjadi kekeliruan atau salah perhitungan.

Untuk mendapatkan hasil perhitungan yang lebih akurat menggunakan algoritma *Resource Request* dan algoritma *Safety*, maka dibuat Sistem Pemodelan Simulasi Permohonan Kredit Menggunakan algoritma Banker untuk dapat mempermudah perhitungan antara jumlah dana pada bank dengan *customer*/nasabah apakah disetujui atau ditunda.

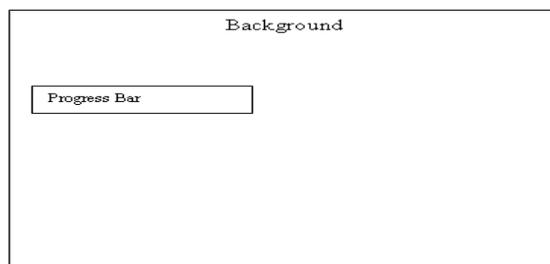
## 3. Rancangan Sistem

### 3.1. Rancangan Output

Adapun rancangan output yang akan dibuat adalah sebagai berikut :

#### 1. Rancangan Splash Screen

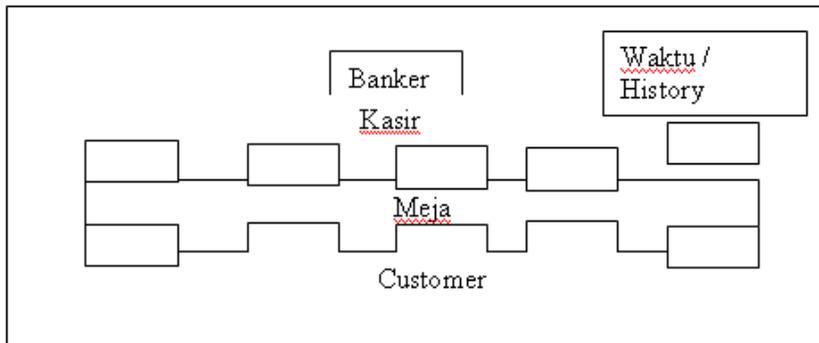
*Form splash screen* merupakan *form* awal yang tampil ketika perangkat lunak dijalankan. *Form splash screen* berisi nama perangkat lunak, nama pembuat dan nama kampus. Rancangan *form splash screen* dapat dilihat pada Gambar 1. berikut :



Gambar 1. Rancangan form splash screen

### 1. Rancangan Form Simulasi

*Form Simulasi* berfungsi sebagai tempat visualisasi simulasi algoritma Banker. Rancangan *form* simulasi dapat dilihat pada Gambar 2. berikut :



Gambar 2. Rancangan form simulasi

### 1. Rancangan Form Note

*Form note* (laporan) merupakan *form* yang menampilkan hasil analisis algoritma terhadap suatu transaksi permohonan kredit dan catatan transaksi yang terjadi. Rancangan *form note* (laporan) dapat dilihat pada Gambar 3. berikut :

History

Gambar 3. Rancangan form note (laporan)

### 1. Rancangan Form About

*Form about* merupakan *form* yang berfungsi untuk menampilkan nama perangkat lunak teori algoritma Banker. Rancangan *form* About dapat dilihat pada Gambar 4. berikut :

Gambar 4. Rancangan form About

### 1.1. Rancangan Input

Adapun rancangan input yang dibuat adalah sebagai berikut :

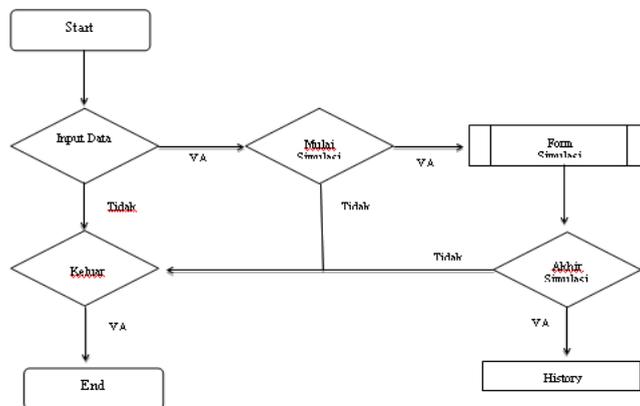
#### 1. Rancangan Form Input Simulasi

*Form input* merupakan *form* yang berfungsi untuk mengatur *input* perangkat lunak. *Input* perangkat lunak berupa banyak tipe *resource* yang dimiliki *bank* (maksimum 5 tipe), nama *resource* (diasumsikan berupa mata uang yang dimiliki bank), besar *resource* yang dimiliki bank (dana yang dimiliki bank) dan mengatur keadaan awal simulasi *customer*. Keadaan awal simulasi berupa batas maksimum dan besar alokasi *resource customer* pada saat simulasi akan dimulai. Keadaan awal *customer* bersifat *optional*, artinya tidak wajib di-*input*. Apabila keadaan awal *customer* tidak di-*input*, maka perangkat lunak menganggap bank berada dalam keadaan kosong pada saat simulasi dimulai, menunggu *customer* masuk ke dalam bank satu per satu. Keadaan berikutnya dihasilkan secara acak oleh komputer. Rancangan *form input* dapat dilihat pada Gambar 5. berikut :

Gambar 5. Rancangan form input

## 1. Logika Program

Ada pun logika dari program yang dirancang adalah sebagai berikut :



Gambar 6. Rancangan Flowchart Program

## HASIL DAN PEMBAHASAN

### 1. Hasil

Adapun tampilan dari hasil perancangan sistem adalah sebagai berikut :

#### a. Tampilan Form Input

Tampilan form input dapat dilihat pada Gambar 7. dibawah ini :

Perangkat Lunak: Simulasi Algoritma Banker

Created by: <NAMA>

Simulasi Banker

### FORM INPUT

A.) Banyak resource yang dimiliki oleh bank = 5 tipe      Lama simulasi = 500 detik

B.) Input nama resource (maks. 3 huruf).      C.) Input besar resource pada bank.

Tipe Resource	Nama Resource	Tipe Resource	Banyak Resource
R1	IDR	R1	100
R2	USD	R2	100
R3	AUS	R3	100
R4	GBP	R4	100
R5	YEN	R5	100

D.) Input jumlah customer (maks. 5 customer). = 5 customer

Input keadaan awal simulasi.      ALOKASI

Customer	MAKSIMUM					ALOKASI				
	R1	R2	R3	R4	R5	R1	R2	R3	R4	R5
Customer - 1										
Customer - 2										
Customer - 3										
Customer - 4										
Customer - 5										

About      Mulai Simulasi Banker      Keluar

Gambar 7. Tampilan Form Input

a. Tampilan Form Awal Simulasi

Tampilan form awal simulasi dapat dilihat pada Gambar 8. dibawah ini :



Gambar 8. Tampilan Form Awal Simulasi

a. Tampilan Form History

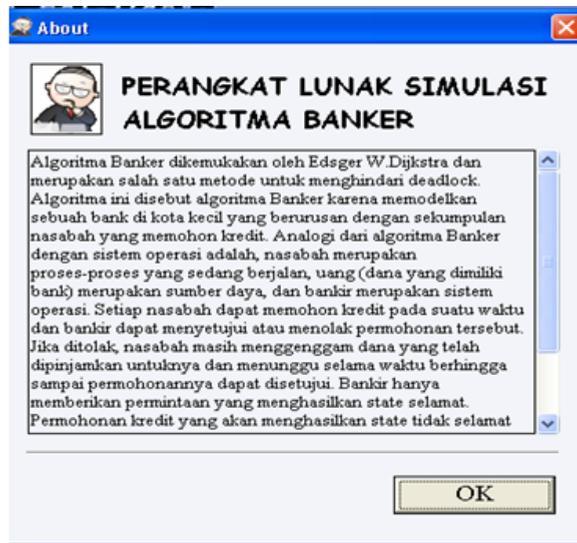
Tampilan form *history* dapat dilihat pada Gambar 9. dibawah ini :

t	Cust	Kredit	Status
12	Cust-1	(3,3,2,1,2)	Cust-1 memohon kredit kepada bankir.
12	Cust-1		Permohonan kredit Cust-1 disetujui (APPROVED).
13	Cust-3	(3,9,7,6,1)	Cust-3 memohon kredit kepada bankir.
13	Cust-3		Permohonan kredit Cust-3 disetujui (APPROVED).
15	Cust-1	(1,7,9,1,1)	Cust-1 memohon kredit kepada bankir.
15	Cust-1		Permohonan kredit Cust-1 disetujui (APPROVED).
15	Cust-2	(22,8,17,13,14)	Cust-2 memohon kredit kepada bankir.
15	Cust-2		Permohonan kredit Cust-2 disetujui (APPROVED).
15	Cust-3	(10,27,8,6,8)	Cust-3 telah mengembalikn pinjamannya.
17	Cust-2	(0,0,1,1,0)	Cust-2 memohon kredit kepada bankir.
17	Cust-2		Permohonan kredit Cust-2 disetujui (APPROVED).
19	Cust-2	(30,13,19,30,18)	Cust-2 telah mengembalikn pinjamannya.
10	Cust-4	(3,16,1,4,1)	Cust-4 memohon kredit kepada bankir.
10	Cust-4		Permohonan kredit Cust-4 disetujui (APPROVED).
12	Cust-1	(1,2,1,4,2)	Cust-1 memohon kredit kepada bankir.
12	Cust-1		Permohonan kredit Cust-1 disetujui (APPROVED).
12	Cust-4	(3,17,1,5,2)	Cust-4 telah mengembalikn pinjamannya.
17	Cust-5	(1,14,1,6,4)	Cust-5 memohon kredit kepada bankir.
17	Cust-5		Permohonan kredit Cust-5 disetujui (APPROVED).
37	Cust-1	(0,0,1,0,0)	Cust-1 memohon kredit kepada bankir.
37	Cust-1		Permohonan kredit Cust-1 disetujui (APPROVED).
39	Cust-1	(5,12,16,12,13)	Cust-1 telah mengembalikn pinjamannya.
44	Cust-5	(0,0,1,2,1)	Cust-5 memohon kredit kepada bankir.
44	Cust-5		Permohonan kredit Cust-5 disetujui (APPROVED).

Gambar 9. Tampilan Form History

a. Tampilan Form About

Tampilan form about dapat dilihat pada Gambar 10. dibawah ini :



Gambar 10. Tampilan Form History

## 1. Pembahasan

### 1.1 Spesifikasi Hardware dan Software

Spesifikasi perangkat keras yang direkomendasikan untuk menjalankan perangkat lunak simulasi ini adalah sebagai berikut :

1. Prosesor *Pentium* IV 1.6 GHz.
2. Memori (RAM) sebesar 128 MB.
3. *Harddisk* dengan minimal *free space* 500 MB.
4. Monitor SVGA dan *VGA Card* 64 MB dengan resolusi minimum 1024 x 768..
5. *Keyboard* dan *Mouse*.

Adapun perangkat lunak (*software*) yang digunakan untuk menjalankan aplikasi ini adalah lingkungan sistem operasi *Microsoft Windows 98, 98 Second Edition (98SE)* atau *Microsoft Windows NT/2000/XP*.

### 1.2 Pengujian Program

Sebagai contoh untuk menguji *output program*, *input* perangkat lunak (tipe *resource*, nama *resource*, besar *resource* bank dan keadaan awal *customer*) simulasi seperti terlihat pada Gambar 11. berikut ini :

Simulasi Algoritma Bankir

FOMITHA BEUNI SHOFFANNY

### FORM INPUT

A.) Banyak resource yang dimiliki oleh bank = 3 tipe Lama simulasi = 500 detik.

B.) Input nama resource (maks. 3 huruf).

Tipe Resource	Nama Resource
R1	USD
R2	USD
R3	A/R

C.) Input besar resource pada bank.

Tipe Resource	Banyak Resource
R1	100
R2	100
R3	100

D.) Input jumlah customer (maks. 5 customer). = 3 customer

Input keadaan awal simulasi.

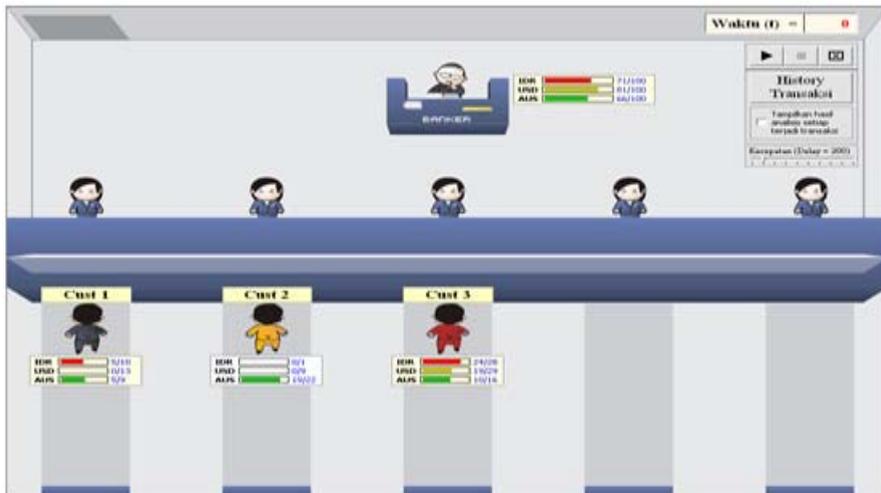
Customer	MAKSIMUM			ALOKASI		
	R1	R2	R3	R1	R2	R3
Customer - 1	10	15	0	5	0	5
Customer - 2	1	0	22	0	0	10
Customer - 3	28	29	10	24	19	10

ACAK

About Mulai Simulasi Bankir Keluar

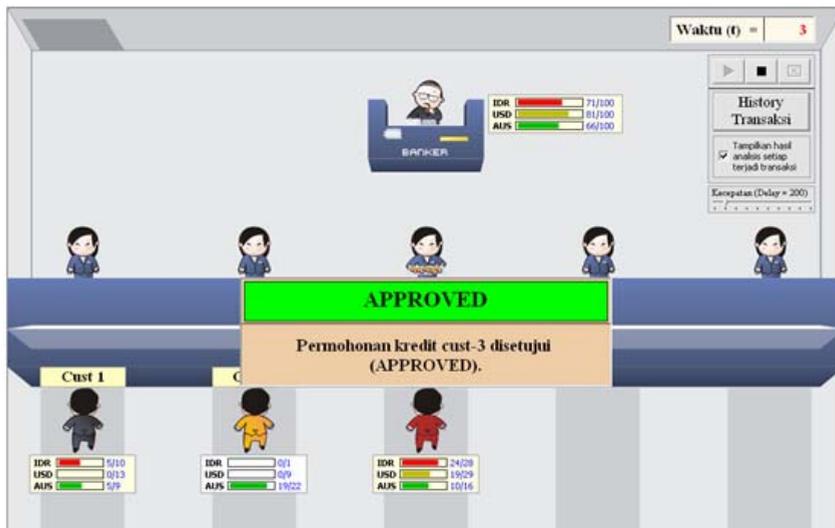
Gambar 11. Tampilan Uji Coba Input perangkat lunak

Tampilan awal simulasi seperti terlihat pada Gambar 12.



Gambar 12. Tampilan Uji Coba awal simulasi

Sebagai contoh, misalkan pada saat  $t = 3$ , customer-3 memohon kredit (3,3, 2) kepada bankir. Tampilan proses seperti terlihat pada Gambar 13.



Gambar 13. Tampilan simulasi saat customer-3 memohon kredit (3,3,2)

Perhitungan eksekusi dengan algoritma banker (*resource request* dan *safety*) adalah:

KEJADIAN:

$t = 3$ , Cust-3 memohon kredit (3,3,2) kepada bankir.

ANALISIS DENGAN RESOURCE REQUEST ALGORITHM:

1. CEK REQUEST HARUS  $\leq$  AVAILABLE.

- $3 \leq 71$  (TRUE)
- $3 \leq 81$  (TRUE)
- $2 \leq 66$  (TRUE)

2. BANKER BERPURA-PURA UNTUK MEMENUHI PERMOHONAN CUSTOMER.

$$\text{IDR}_{\text{Bank}} = \text{IDR}_{\text{Bank}} - \text{IDR}_{\text{Request}} = 71 - 3 = 68$$

$$\text{Allocation}_{\text{IDR}_{\text{Cust3}}} = \text{Allocation}_{\text{IDR}_{\text{Cust3}}} + \text{IDR}_{\text{Request}}$$

$$= 24 + 3 = 27$$

$$\text{Needs\_IDR\_Cust3} = \text{Needs\_IDR\_Cust3} - \text{IDR\_Request} = 4 - 3 = 1$$

$$\text{USD\_Bank} = \text{USD\_Bank} - \text{USD\_Request} = 81 - 3 = 78$$

$$\begin{aligned} \text{Allocation\_USD\_Cust3} &= \text{Allocation\_USD\_Cust3} + \text{USD\_Request} \\ &= 19 + 3 = 22 \end{aligned}$$

$$\text{Needs\_USD\_Cust3} = \text{Needs\_USD\_Cust3} - \text{USD\_Request} = 10 - 3 = 7$$

$$\text{AUS\_Bank} = \text{AUS\_Bank} - \text{AUS\_Request} = 66 - 2 = 64$$

$$\begin{aligned} \text{Allocation\_AUS\_Cust3} &= \text{Allocation\_AUS\_Cust3} + \text{AUS\_Request} \\ &= 10 + 2 = 12 \end{aligned}$$

$$\text{Needs\_AUS\_Cust3} = \text{Needs\_AUS\_Cust3} - \text{AUS\_Request} = 6 - 2 = 4$$

### 3. PERIKSA APAKAH KEADAAN SAFE DENGAN SAFETY ALGORIGHTM

Selesaikan kredit cust-1.

$$\text{IDR\_Bank} = \text{IDR\_Bank} + \text{Allocation\_IDR\_Cust1} = 68 + 5 = 73$$

$$\text{USD\_Bank} = \text{USD\_Bank} + \text{Allocation\_USD\_Cust1} = 78 + 0 = 78$$

$$\text{AUS\_Bank} = \text{AUS\_Bank} + \text{Allocation\_AUS\_Cust1} = 64 + 5 = 69$$

Finish(Cust1) = TRUE.

Selesaikan kredit cust-2.

$$\text{IDR\_Bank} = \text{IDR\_Bank} + \text{Allocation\_IDR\_Cust2} = 73 + 0 = 73$$

$$\text{USD\_Bank} = \text{USD\_Bank} + \text{Allocation\_USD\_Cust2} = 78 + 0 = 78$$

$$\text{AUS\_Bank} = \text{AUS\_Bank} + \text{Allocation\_AUS\_Cust2} = 69 + 19 = 88$$

Finish(Cust2) = TRUE.

Selesaikan kredit cust-3.

$$\text{IDR\_Bank} = \text{IDR\_Bank} + \text{Allocation\_IDR\_Cust3} = 73 + 27 = 100$$

$$\text{USD\_Bank} = \text{USD\_Bank} + \text{Allocation\_USD\_Cust3} = 78 + 22 = 100$$

$$\text{AUS\_Bank} = \text{AUS\_Bank} + \text{Allocation\_AUS\_Cust3} = 88 + 12 = 100$$

Finish(Cust3) = TRUE.

Semua kredit terselesaikan (SAFE STATE).

Oleh karena itu, permohonan kredit cust-3 disetujui (APPROVED).

Pada saat  $t = 5$ , *customer-1* memohon kredit (30,11,20,26,1) kepada bankir dan permohonan kredit ditunda. Tampilan proses seperti terlihat pada Gambar 14. berikut:



Gambar 14. Tampilan simulasi saat *customer-1* memohon kredit (30,11,20,26,1)

Perhitungan eksekusi dengan algoritma banker (*resource request* dan *safety*) adalah:

KEJADIAN:

$t = 5$ , Cust-1 memohon kredit (30,11,20,26,1) kepada bankir.

ANALISIS DENGAN RESOURCE REQUEST ALGORITHM:

1. CEK REQUEST HARUS  $\leq$  AVAILABLE.

-  $30 \leq 10$  (FALSE)

Permintaan lebih besar dari dana yang tersedia.

Permohonan kredit Cust-1 ditunda (PENDING).

Tampilan *history* perangkat lunak seperti terlihat pada Gambar 15. berikut :

t	Cust	Kredit	Status
13	Cust-3	(3,3,2)	Cust-3 memohon kredit kepada bankir.
13	Cust-3		Permohonan kredit Cust-3 disetujui (APPROVED).
18	Cust-3	(1,5,3)	Cust-3 memohon kredit kepada bankir.
18	Cust-3		Permohonan Kredit Cust-3 disetujui (APPROVED).
11	Cust-3	(0,1,1)	Cust-3 memohon kredit kepada bankir.
11	Cust-3		Permohonan Kredit Cust-3 disetujui (APPROVED).
12	Cust-1	(5,7,3)	Cust-1 memohon kredit kepada bankir.
12	Cust-1		Permohonan kredit Cust-1 disetujui (APPROVED).
12	Cust-2	(1,9,3)	Cust-2 memohon kredit kepada bankir.
12	Cust-2		Permohonan kredit Cust-2 disetujui (APPROVED).
14	Cust-2	(1,9,22)	Cust-2 telah mengesahkan pinjamannya.
17	Cust-1	(0,6,1)	Cust-1 memohon kredit kepada bankir.
17	Cust-1		Permohonan kredit Cust-1 disetujui (APPROVED).
19	Cust-1	(10,13,9)	Cust-1 telah mengesahkan pinjamannya.
26	Cust-3	(0,1,0)	Cust-3 memohon kredit kepada bankir.
26	Cust-3		Permohonan Kredit Cust-3 disetujui (APPROVED).
28	Cust-3	(28,29,16)	Cust-3 telah mengesahkan pinjamannya.

Gambar 15. Tampilan Hasil Uji Coba history perangkat lunak

### 1.1 Kelebihan Dan Kekurangan Sistem Yang Dirancang

Adapun kelebihan dari sistem yang dirancang adalah sebagai berikut :

1. Perhitungan menggunakan algoritma *Resource Request* dan algoritma *Safety* lebih akurat.
2. Hasil perhitungan dapat ditampilkan lebih cepat.
3. Dapat menampilkan hasil dari customer yang disetujui ataupun yang ditunda tanpa menghitung antar satu-persatu customer.

Selain memiliki kelebihan sistem yang dirancang masih terdapat kekurangan diantaranya sebagai berikut :

1. Hasil perhitungan belum dapat disimpan secara langsung kedalam bentuk file tertentu misalnya .txt atau .doc.
2. Batas dana bank yang dimiliki banker hanya bernilai 100.
3. *Customer* dan banyaknya mata uang yang dipinjamkan hanya 5.

## KESIMPULAN

Adapun simpulan yang penulis buat adalah sebagai berikut:

1. Perangkat lunak juga menjelaskan dan menampilkan hasil analisis transaksi dengan menggunakan algoritma Banker, sehingga dapat membantu pemahaman terhadap cara kerja algoritma *Banker* dalam mata kuliah sistem operasi.
2. Perangkat lunak simulasi ini merupakan ilustrasi dari proses-proses yang berkompetisi untuk meminta sumber daya kepada sistem operasi.
3. Algoritma Banker adalah suatu metode untuk mencegah *deadlock*. Namun, algoritma Banker tidak pernah diterapkan dalam sistem operasi yang sebenarnya, karena algoritma Banker membatasi jumlah *resource* maksimum untuk setiap proses, sedangkan proses-proses biasanya belum dapat mengetahui jumlah *resource* maksimum yang dibutuhkannya ketika diciptakan.

## DAFTAR PUSTAKA

1. Hariyanto, B, 1999, *Sistem Operasi*, Edisi 2, Informatika, Bandung.
2. Hariyanto, B, 1999, *Sistem Operasi Lanjut*, Informatika, Bandung.
3. James A, O'Brien, 2005, *Sistem Informasi*, Jakarta.
4. Kusumadewi, S, 2000, *Sistem Operasi*, Edisi2, Graha Ilmu, Jakarta.
5. Novian, A dan Andi, 2004, *Panduan MS. Visual Basic 6*, Yogyakarta.
6. Ramadhan, A, 2004, *MS. Visual Basic 6 (Seri Penuntun Praktis)*, PT. Elex Media Komputindo, Jakarta.
7. Supardi, Y, 2006, *Microsoft Visual Basic 6.0 Untuk Segala Tingkat*, PT. ElexMedia Komputindo, Jakarta.
8. Sandi, S, 1991, *Simulasi Teknik Program dan Metode Analisis*, Andi Offset, Yogyakarta, Indonesia.
9. Tata Subtari, 2004, *Sistem Informasi Komputer*, Yogyakarta
10. <http://www.rsu.edu/faculty/PMacpherson/Programs/banker.html>.
11. [http://en.wikipedia.org/wiki/Banker's\\_algorithm.html](http://en.wikipedia.org/wiki/Banker's_algorithm.html).
12. <http://www.algana.co.uk/Algorithms/OperatingSystems/Banker/Banker.htm>
13. <http://www.if.uidaho.edu/~bgray/classes/cs341/doc/banker.html>.
14. <http://www.buchli.org/frank/work/os/deadlock.html>.
15. <http://www.ilmukomputer.com/umum/ibam/ibam-os-html.zip>.