A

Very nice report.

SOLUTION OF UNSTEADY FLOW IN OPEN CHANNELS BY METHOD OF CHARACTERISTICS

Djoko Luknanto and M.B. Sharifi

Term Project, 14 December 1988

THE HARTREE FIXED-GRID METHOD

Djoko Luknanto

Term Project, 14 December 1988

I. OBJECTIVES

To solve unsteady flow in a single channel using method of characteristics. Two kinds of method of characteristic are used, the first one is "Hartree" method (a fixed grid method) and the second is variable grid method.

II. INTRODUCTION

Theoretically method of characteristic is the most promising method to solve unsteady flow in open channels. Using this method the partial diff eqns of the governing equations are reduced into a system of ordinary differential eqns. A system of ode's is much ~~more~~ easier to handle than a system of pde's, furthermore one can find the "exact" solution of a system of ode's as long as the integration evaluation is solve exactly.

III. THE GOVERNING EQUATIONS

Continuity:

$$\frac{\partial A}{\partial t} + \frac{\partial Q}{\partial x} = 0 \qquad (1)$$

Dynamic:

$$\frac{\partial Q}{\partial t} + \frac{\partial}{\partial x}\left(\frac{Q^2}{A}\right) + gA\frac{\partial y}{\partial x} + gAS_f = 0 . \qquad (2)$$

- Need to work with unknowns : $u(x,t)$ and $h(x,t)$.

Recall : $Q = u(x,t) * A(h)$.

$$\frac{\partial Q}{\partial x} = u\frac{\partial A}{\partial x} + A\frac{\partial u}{\partial x} = u\left[\frac{\partial A}{\partial h}\frac{\partial h}{\partial x} + \underbrace{\left(\frac{\partial A}{\partial x}\right)_{h=\text{const}}}_{E}\right] + A\frac{\partial u}{\partial x}$$

$$= u\left[b\frac{\partial h}{\partial x} + E\right] + A\frac{\partial u}{\partial x} .$$

$$\frac{\partial A}{\partial t} = \frac{\partial A}{\partial h}\cdot\frac{\partial h}{\partial t} = b\frac{\partial h}{\partial t}$$

$$\frac{\partial Q}{\partial t} = u\frac{\partial A}{\partial t} + A\frac{\partial u}{\partial t} = -u\frac{\partial Q}{\partial x} + A\frac{\partial u}{\partial t} .$$

$$\frac{\partial (Q^2/A)}{\partial x} = \frac{2Q}{A}\frac{\partial Q}{\partial x} - \frac{Q^2}{A^2}\frac{\partial A}{\partial x} = 2u\frac{\partial Q}{\partial x} - u\left[\frac{\partial Q}{\partial x} - A\frac{\partial u}{\partial x}\right] = u\frac{\partial Q}{\partial x} + uA\frac{\partial u}{\partial x}$$

Subst. into (1) & (2)

Continuity : $\frac{\partial h}{\partial t} + \frac{A}{b}\frac{\partial u}{\partial x} + u\frac{\partial h}{\partial x} + \frac{uE}{b} = 0$ (3)

Dynamic : $\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} + g\frac{\partial (z+h)}{\partial x} + gS_f = 0$.

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} + g\frac{\partial h}{\partial x} + g(S_f - S_0) = 0 \qquad (4).$$

Let the celerity of the flow $c = \sqrt{\frac{gA}{b}}$ or $c^2 = \frac{gA}{b}$.

$$2c\frac{\partial c}{\partial t} = g\frac{\partial h}{\partial t} \quad \text{and} \quad 2c\frac{\partial c}{\partial x} = g\left[\frac{b\frac{\partial A}{\partial h}\frac{\partial h}{\partial x} - A\frac{\partial b}{\partial h}\frac{\partial h}{\partial x}}{b^2}\right] = g\underbrace{\left(1-\frac{A}{b^2}\frac{\partial b}{\partial h}\right)}_{F}\frac{\partial h}{\partial x}$$

Subst. into (3) & (4)

$$\frac{2c}{g}\frac{\partial c}{\partial t} + \frac{A}{b}\frac{\partial u}{\partial x} + \frac{2uc}{gF}\frac{\partial c}{\partial x} + \frac{uE}{b} = 0$$

$$2\frac{\partial c}{\partial t} + c\frac{\partial u}{\partial x} + \frac{2u}{F}\frac{\partial c}{\partial x} + \frac{guE}{bc} = 0 \qquad (5)$$

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} + \frac{2c}{F}\frac{\partial c}{\partial x} + g(S_f - S_0) = 0 \qquad (6).$$

$$(6)+\lambda(5) = \frac{\partial}{\partial t}(u+\lambda 2c) + (u+\lambda c)\frac{\partial u}{\partial x} + \left(\frac{u}{F} + \frac{c}{\lambda F}\right)\frac{\partial 2\lambda c}{\partial x} + \frac{\lambda guE}{bc} + g(S_f - S_0) = 0$$

Need to transfor the above eqn. into ode's :

Let $u + \lambda c = \frac{u}{F} + \frac{c}{\lambda F}$, the rewrite the eqn

$$\frac{\partial}{\partial t}(u+\lambda 2c) + (u+\lambda c)\frac{\partial}{\partial x}(u+\lambda 2c) + \frac{\lambda guE}{bc} + g(S_f - S_0) = 0$$

Then we can write : $\boxed{\frac{D}{Dt}(u+\lambda 2c) + \frac{\lambda guE}{bc} + g(S_f - S_0) = 0}$ (6a)

if : $\boxed{\frac{dx}{dt} = u + \lambda c}$. (6.b)

Now what is the value of λ ?

$$u + \lambda c = \frac{u}{F} + \frac{c}{\lambda F} \rightarrow \lambda F u + \lambda^2 F c - \lambda u - c = 0.$$

$$(Fc)\lambda^2 + (Fu - u)\lambda - c = 0$$

$$\boxed{\lambda_{1,2} = \frac{u - Fu \pm \sqrt{(u-Fu)^2 + 4Fc^2}}{2Fc}} \qquad (6.c)$$

Recall that $F = 1 - \frac{A}{b^2}\frac{\partial b}{\partial h}$.

NOTE : If $\frac{\partial b}{\partial h} = 0$ where b = water surface width
h = water depth
then $F = 1$ and $\lambda = \pm 1$

From now on assumed $\frac{\partial b}{\partial h} = 0$, then the governing eqns become:

Positive characteristic:

$$\left.\begin{aligned} &\frac{D}{Dt}(u+2c) = g(S_0 - S_f) - \frac{guE}{bc} \\ &\text{along } \frac{dx}{dt} = u+c \end{aligned}\right\} \qquad (7)$$

Negative characteristic:

$$\left.\begin{aligned} &\frac{D}{Dt}(u-2c) = g(S_0 - S_f) + \frac{guE}{bc} \\ &\text{along } \frac{dx}{dt} = u-c \end{aligned}\right\} \qquad (8)$$

where : u = velocity of flow.

c = wave celerity $= \sqrt{\frac{gA}{b}}$

g = gravity.

A = wetted area

b = water-surface width

S_0 = bed slope $= \frac{-dz}{dx}$, z = bed elevation

S_f = friction slope.

$= \frac{n^2u^2}{R^{4/3}}$ in metric system

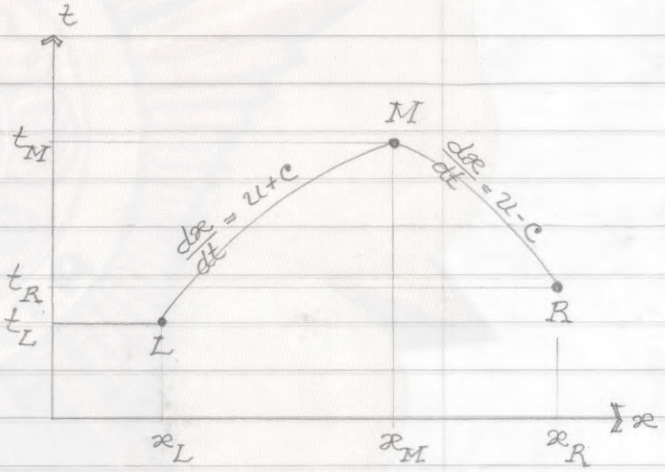
n = Manning coefficient

R = hydraulic radius $= A/P$

P = wetted perimeter

$E = \left(\frac{\partial A}{\partial x}\right)_{h=\text{const}}$

h = waterdepth.



With reference to the figure, solution of eqns (7) & (8) is as follows:

$$x_M - x_L = \int_{t_L}^{t_M} (u+c)\,dt \qquad (9)$$

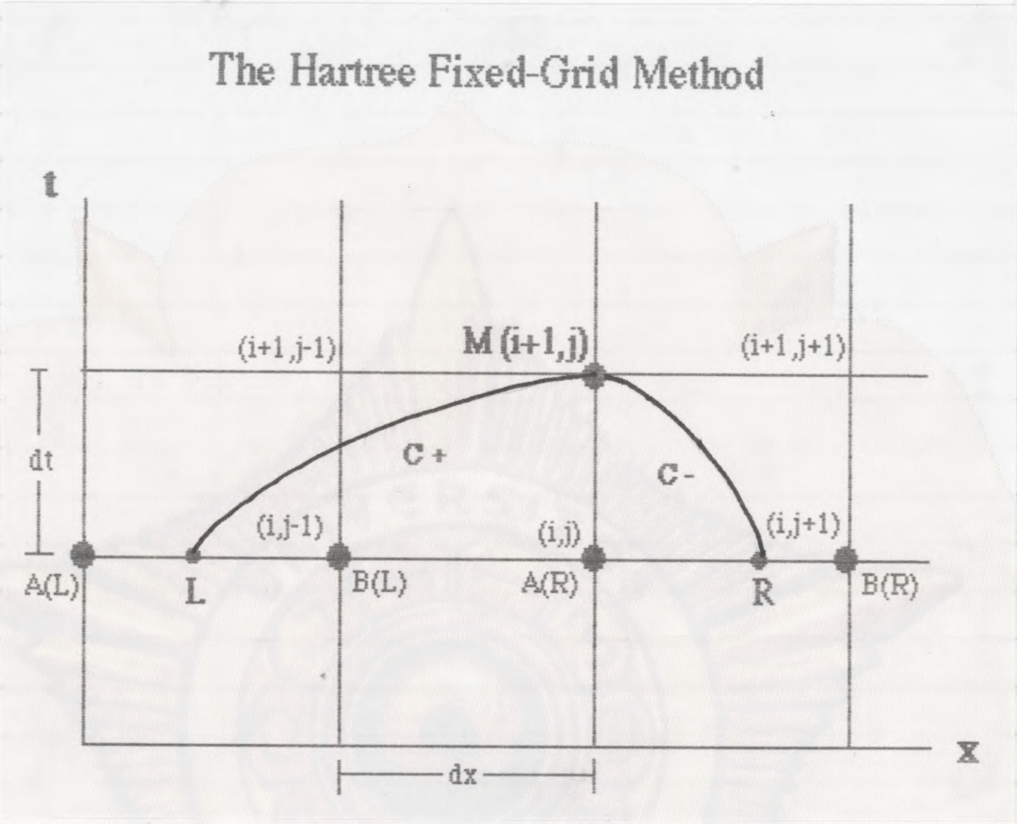
$$(u+2c)_M - (u+2c)_L = g\int_{t_L}^{t_M} \left[S_0 - S_f - \frac{uE}{bc}\right] dt \qquad (10)$$

$$x_M - x_R = \int_{t_L}^{t_M} (u-c)\,dt \qquad (11)$$

$$(u-2c)_M - (u-2c)_R = g\int_{t_L}^{t_M} \left[S_0 - S_f + \frac{uE}{bc}\right] dt \qquad (12)$$

IV. FIXED-GRID METHOD .

As mentioned in the objective of this paper. The so called "Hartree" method uses a fixed-grid x-t domain as described below.



By using the above grid. the governing ode's ; eqns (9) - (12) can be approximated as :

$$x_M = x_L + \tfrac{1}{2}\Delta t\,(u_L + c_L + u_M + c_M) \quad (13)$$

$$u_M + 2c_M = u_L + 2c_L + \tfrac{1}{2}g\Delta t\left[(S_0 - S_f - \frac{uE}{bc})_L + (S_0 - S_f - \frac{uE}{bc})_M\right] \quad (14)$$

C^+

$$x_M = x_R + \tfrac{1}{2}\Delta t\,(u_R - c_R + u_M - c_M) \quad (15)$$

$$u_M - 2c_M = u_R - 2c_R + \tfrac{1}{2}g\Delta t\left[(S_0 - S_f + \frac{uE}{bc})_R + (S_0 - S_f + \frac{uE}{bc})_M\right] \quad (16)$$

C^-

The unknowns are $(u,h)_M$, $(u,h)_L$, $(u,h)_R$, x_L, x_R = 8 unknowns. So one need 4 more eqns in order to solve eqns (13) - (16). The eqns are. the interpolation formulas for $(u,h)_L$ and $(u,h)_R$. In this paper linear interpolations are used to compute $(u,h)_{L\,and\,R}$

The values at point L are interpolated using points A(L) and B(L). the values at point R are interpolated using points A(R) and B(R)

The interpolation formulas :

$$u_L = u_{A_L} + \frac{x_L - x_{A_L}}{\Delta x}\left(u_{B_L} - u_{A_L}\right) \quad (17).$$

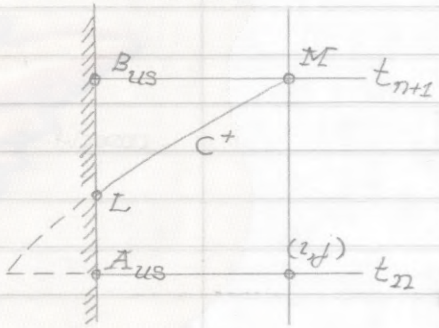
$$h_L = h_{A_L} + \frac{x_L - x_{A_L}}{\Delta x}\left(h_{B_L} - h_{A_L}\right) \quad (18).$$

$$u_R = u_{A_R} + \frac{x_R - x_{A_R}}{\Delta x}\left(u_{B_R} - u_{A_R}\right) \quad (19)$$

$$h_R = h_{A_R} + \frac{x_R - x_{A_R}}{\Delta x}\left(h_{B_R} - h_{A_R}\right) \quad (20).$$

If the trajectories of characteristics exceed the boundary then one might use another type of interpolation. In all the interpolations given below, it is assumed the values of (u,h) at the nodal values can be obtained directly or indirectly from the boundary conditions

Along the trajectory of C^+ : $\frac{dx}{dt} = u + c$



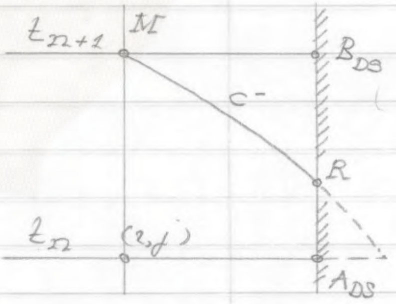
$$x_M - x_L = \tfrac{1}{2}(t_M - t_L)(u_L + c_L + u_M + c_M).$$

$$t_L = t_M - \frac{2\Delta x}{u_L + c_L + u_M + c_M} \quad (21)$$

$$u_L = u_{B_{US}} - \frac{t_{n+1} - t_L}{\Delta t}\left(u_{B_{US}} - u_{A_{US}}\right) \quad (22).$$

$$h_L = h_{B_{US}} - \frac{t_{n+1} - t_L}{\Delta t}\left(h_{B_{US}} - h_{A_{US}}\right) \quad (23)$$

Along the trajectory of C^- : $\frac{dx}{dt} = u - c$



$$x_M - x_R = \tfrac{1}{2}(t_M - t_R)(u_R - c_R + u_M - c_M)$$

$$t_R = t_M + \frac{2\Delta x}{u_R - c_R + u_M - c_M} \quad (24)$$

$$u_R = u_{B_{DS}} - \frac{t_{n+1} - t_R}{\Delta t}\left(u_{B_{DS}} - u_{A_{DS}}\right) \quad (25)$$

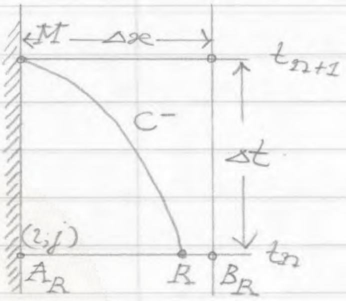
$$h_R = h_{B_{DS}} - \frac{t_{n+1} - t_R}{\Delta t}\left(h_{B_{DS}} - h_{A_{DS}}\right) \quad (26)$$

V. THE ALGORITHM

V.1. Upstream Boundary Condition.

a) Waterdepth is given as a function of t : $h = h(t)$.

1. Compute $h_M = h(t = t_{n+1})$.
2. Compute $c_M = \sqrt{(gA/b)_M}$ } (27).



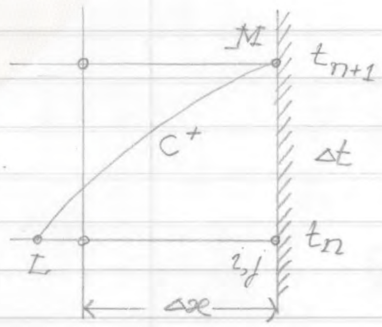
2. Initialize $u_M = u_{(i,j)}$.
3. Approximate eqn (15) as $x_R = x_M - \Delta t(u_M - c_M)$ (28)
4. Compute $(u,h)_R$ using eqns (19) & (20).
5. Compute u_M using eqn (16) which is a quadratic eqn wrt u_M
6. Compute x_R using eqn (15).
7. Repeat step (4) to (6) until desired accuracy is achieved.

b) Discharge is given as a function of t : $Q = Q(t)$.

1. Initialize $(u,h)_M = (u,h)_{i,j}$, $Q_M = Q(t = t_{n+1})$.
2. Compute x_R using eqn (28).
3. Compute $(u,h)_R$ using eqn (19) & (20).
4. Substitute $u_M = Q_M / A_M$ into eqn (16) and rewrite as $f_1(h_M) = 0$. (29).
5. Solve $f_1(h_M) = 0$ to get h_M using Newton - Raphson method.
6. Compute $u_M = Q_M / A_M$
7. Compute x_R using eqn. (15).
8. Repeat step (3) to (7) until desired accuracy is achieved

V.2. Downstream Boundary Condition.

Downstream boundary condition is given as locally uniform flow, meaning $S_f \approx S_0$.



1. Initialize $(u,h)_M = (u,h)_{i,j}$
2. Eqn (13) is approx. as $x_L = x_M - \Delta t\,(u_M + c_M)$. (30)
3. Compute $(u,h)_L$ using eqns (17) & (18).
4. Substitute $u_M = \frac{1}{n} R^{2/3} S_0^{\,1/2}$ into eqn. (14) and rewrite as $f_2(h_M) = 0$. (31)
5. Solve $f_2(h_M) = 0$ for h_M using Newton - Raphson method.
6. Compute $u_M = \frac{1}{n} R^{2/3} S_0^{\,1/2}$.
7. Compute x_L using eqn. (13).
8. Repeat step (3) to (7) until desired accuracy is achieved.

V.3 Interior Points.

1. Initialize $(u,h)_M = (u,h)_{i,j}$
2. Approximate x_L and x_R using eqns (30) & (28).
3. Compute $(u,h)_L$ and $(u,h)_R$ using eqns (17) - (20).
4. Solve $(u,h)_M$ by solving both eqns. (14) & (16). Rewrite eqns.(14) and (16) as

Eqn (14): $f_3(h_M, u_M) = 0$ (32)

(16): $f_4(h_M, u_M) = 0$ (33)

Solve using Newton-Raphson technique to get $(u,h)_M$

5. Compute x_L and x_R using eqns. (13) & (15).
6. Repeat step (3) to (5) until desired accuracy is achieved.

VI. NEWTON-RAPHSON TECHNIQUE

Since the algorithms on section V above are heavily based on Newton-Raphson technique, so the technique and the eqns which are used will be discussed in detail in this section.

VI.1 Upstream Boundary Condition

a) $h = h(t)$ as an u/s b.c :

See Section V.1.a, step 5 :

Eqn. (16) : $$\underbrace{u_R - 2C_R + \tfrac{1}{2}g\Delta t\left(S_0 - S_f + \frac{uE}{bc}\right)_R}_{GM_R} + \tfrac{1}{2}g\Delta t\left(S_0 - S_f + \frac{uE}{bc}\right)_M - u_M + 2C_M = 0.$$

Recall: $S_f = \dfrac{n^2u^2}{R^{4/3}}$. subst. into the eqn.

$$\underbrace{2C_M + GM_R + \tfrac{1}{2}g\Delta t S_0}_{Z_3} \underbrace{- \tfrac{1}{2}g\Delta t n^2 R_M^{-4/3}}_{Z_1} u_M^2 + \underbrace{\left[\left(\tfrac{1}{2}g\Delta t\frac{E}{bc}\right)_M - 1\right]}_{Z_2} u_M = 0.$$

$$\therefore u_M = \frac{-Z_2 - \sqrt{Z_2^2 - 4Z_1Z_3}}{2Z_1} \qquad (34)$$

b) $Q = Q(t)$ as an u/s b.c .

See Section V.1.b, step 5 :

Eqn (16) : $$\underbrace{GM_R + \tfrac{1}{2}g\Delta t S_0}_{Z_3} \underbrace{- \tfrac{1}{2}g\Delta t n^2 Q_M^2}_{Z_1} R_M^{-4/3} A_M^{-2} + \underbrace{\left[\tfrac{1}{2}g\Delta t\left(\frac{E}{bc}\right)_M - 1\right] Q_M}_{Z_2} A_M^{-1} + 2C_M = 0$$

Eqn (29) becomes : $$f_1(h_M) \approx Z_1 R_M^{-4/3} A_M^{-2} + Z_2 A_M^{-1} + 2C_M + Z_3 = 0 \qquad (35)$$

$$\frac{\partial f_1}{\partial h_M} = -z_1\left[\frac{4}{3}A^{-2}R^{-(4/3+1)}\frac{\partial R}{\partial h} + 2A^{-3}R^{-4/3}\frac{\partial A}{\partial h}\right]_M - z_2 A_M^{-2} + A_M^{-1}\frac{\partial z_2}{\partial h_M} + 2\frac{\partial c}{\partial h_M}$$

$$f_1' = -z_1 A^{-2}R^{-4/3}\underbrace{\left[\frac{4}{3}R^{-1}\frac{\partial R}{\partial h} + 2A^{-1}\frac{\partial A}{\partial h}\right]}_{z_5} - z_2 A_M^{-2} + \underbrace{\frac{1}{2}g\Delta t Q A_M^{-1}\left[\frac{\frac{\partial E}{\partial h}c - \frac{\partial c}{\partial h}E}{bc^2}\right]}_{z_4} + 2\frac{\partial c}{\partial h}$$

$$\therefore f_1' = \frac{-1}{A_M^2}\left(\frac{z_1 z_5}{R_M^{4/3}} + z_2\right) + z_4 + 2\left(\frac{\partial c}{\partial h}\right)_M \qquad (36)$$

$$\therefore h_M = h_M - \frac{f_1}{f_1'} \qquad (37).$$

VI. 2. Downstream Boundary Condition.

See Section V. 2, step 4:

Eqn (31):

$$\underbrace{u_L + 2c_L + \frac{1}{2}g\Delta t\left(S_0 - S_f - \frac{uE}{bc}\right)_L}_{GM_L} - 2c_M - \underbrace{\left[\frac{1}{2}g\Delta t\left(\frac{E}{bc}\right)_M + 1\right]}_{z_1} u_M = 0$$

Recall that $u = u(h) = \frac{1}{n}R^{2/3}S_0^{1/2}$.

$$\therefore f_2(h_M) \equiv GM_L - 2c_M - z_1 u_M = 0 \qquad (38)$$

$$\frac{\partial f_2}{\partial h_M} = -2\left(\frac{\partial c}{\partial h}\right)_M - z_1\left(\frac{\partial u}{\partial h}\right)_M - u_M\frac{\partial z_1}{\partial h}$$

$$f_2' = -2\underbrace{\left(\frac{\partial c}{\partial h}\right)_M}_{z_2} - z_1\underbrace{\left(\frac{\partial u}{\partial h}\right)_M}_{z_3} - u_M\underbrace{\frac{1}{2}g\Delta t\left[\frac{\frac{\partial E}{\partial h}c - \frac{\partial c}{\partial h}E}{bc^2}\right]_M}_{z_4} \qquad (39).$$

$$\therefore h_M = h_M - \frac{f_2}{f_2'} \qquad (40).$$

VI. 3. Interior Points.

See Section V. 3, step 4

Add (14) to (16): $2u_M = GM_L + GM_R + g\Delta t\,(S_0 - S_f)_M$.

$$u_M = \underbrace{\frac{1}{2}(GM_L + GM_R) + \frac{1}{2}g\Delta t S_0}_{z_2}\ \underbrace{-\frac{1}{2}g\Delta t\, n^2 R^{-4/3}}_{z_1} u_M^2$$

$$\therefore f_3(h_M, u_M) \equiv z_1 R_M^{-4/3}|u_M|u_M - u_M + z_2 = 0 \qquad (41).$$

Substract (16) from (14): $4c_M = GM_L - GM_R - g\Delta t\left(\frac{uE}{bc}\right)_M$

$$c_M = \underbrace{\frac{1}{4}(GM_L - GM_R)}_{z_3} - \underbrace{\frac{1}{4}g\Delta t\frac{1}{b}}_{z_4}\left(\frac{E}{c}\right)_M u_M$$

$$\therefore f_4(h_M, u_M) = z_3 - z_4\left(\frac{E}{c}\right)_M u_M - c_M \qquad (42).$$

$$\frac{\partial f_3}{\partial h_M} \approx -\frac{4}{3} z_1 R_M^{-(\frac{4}{3}+1)} |u_M| u_M \left(\frac{\partial R}{\partial h}\right)_M \rightarrow a11 \qquad (43)$$

$$\frac{\partial f_3}{\partial u_M} = 2 z_1 R_M^{-4/3} |u_M| - 1 \rightarrow a12 \qquad (44).$$

$$\frac{\partial f_4}{\partial h_M} = - z_4 u_M \left[\frac{\frac{\partial E}{\partial h} c - \frac{\partial c}{\partial h} E}{c^2} \right]_M - \left(\frac{\partial c}{\partial h}\right)_M \rightarrow a21. \qquad (45)$$

$$\frac{\partial f_4}{\partial u_M} = - z_4 \left(\frac{E}{c}\right)_M \rightarrow a22 \qquad (46).$$

Using Newton-Raphson Technique:

$$\begin{Bmatrix} h \\ u \end{Bmatrix}_{M_{New}} = \begin{Bmatrix} h \\ u \end{Bmatrix}_{M_{old}} - \frac{[F]_{M_{old}}}{[F']_{M_{old}}}$$

$$\begin{bmatrix} F' \end{bmatrix}_{M_{old}} \left[\begin{Bmatrix} h \\ u \end{Bmatrix}_{M_{old}} - \begin{Bmatrix} h \\ u \end{Bmatrix}_{M_{new}} \right] = \begin{bmatrix} F \end{bmatrix}_{M_{old}}.$$

or rewrite $[F']_{M_{old}} \begin{Bmatrix} \Delta h \\ \Delta u \end{Bmatrix} = [F]_{M_{old}}$.

$$\therefore h_{M_{new}} = h_{M_{old}} + \Delta h \quad \text{and} \quad u_{M_{new}} = u_{M_{old}} + \Delta u \qquad (47).$$

where Δh & Δu are solved as follows

$$\begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \begin{Bmatrix} \Delta h \\ \Delta u \end{Bmatrix} = \begin{Bmatrix} f_3(h_M, u_M) \\ f_4(h_M, u_M) \end{Bmatrix}$$

Let $\det = a_{11} a_{22} - a_{12} a_{21}$ (48)

then

$$\Delta h = \frac{a_{22} f_3 - a_{12} f_4}{\det} \quad \text{and} \quad \Delta u = \frac{a_{11} f_4 - a_{21} f_3}{\det} \qquad (49).$$

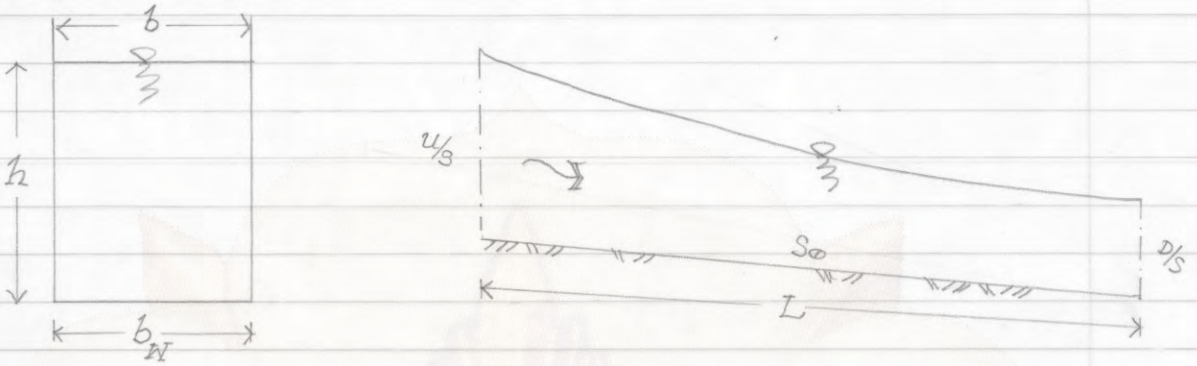
VII. TEST CASE.

The algorithms described above are coded in FORTRAN 77 and then applied to a hypothetical channel described below. For the purpose of evaluation the channel is chosen to be ~~a~~ rectangular in cross ~~sections~~ section (recall that the working eqns are derived with the assumption $\frac{\partial b}{\partial h} = 0$), but the width of bottom of the channel varies along the channel. ✓

Although the channel seems to be a very simple one, ~~but~~ as a test case it is a useful one. By making the bottom width of the channel ~~varies~~ vary the term $\left(\frac{\partial A}{\partial x}\right)_{h=const}$ cannot be neglected.

The geometry of the channel and the boundary conditions are explained in the next section.

VII.1 Geometry of the Channel



b = water surface width
h = water depth
b_N = bottom width
L = length of channel
S_0 = bed slope

Channel data :		
	L	= 24 km.
	Simulation time	= 7 hrs
	Manning coefficient	= $\frac{1}{15}$
	S_0 (constant)	= 0.0005
	u/s bottom elev	= 12 m.
	u/s b_N	= 8 m
	D/S b_N	= 20 m
	Gravity acc.	= 9.81 m/sec^2

VII.2 Initial Conditions

Initial conditions of the flow are given as a prescribed discharge and water-surface elevation.

Discharge : $Q_0 = 100 \ m^3/s$

Water surface el. $y = 23.741 - 0.9653x + 0.0097x^2$ with $x = 0$ at u/s end (x in km, y in m).

VII.3 Upstream Boundary Condition

Upstream b.c is given either as a water depth hydrograph or a discharge hydrograph.

For the Hartree fixed grid method a discharge hydrograph is given, and for variable grid method a water depth hydrograph is used.

- Discharge hydrograph :

Time (min)	≤30	45	60	75	90	105	120	≥135
Q (m^3/s)	100	250	350	300	250	200	150	100

- Waterdepth hydrograph :
Since the record of this hydrograph is very long, so the record is not written here, but can be found in "the result" section.

VII. 4 Downstream Boundary Condition. not always!

To resemble what is done in practice, the d/s boundary condition is taken as locally uniform flows ie S_f is assume to be equal S_0.

VIII. METHOD OF EVALUATION

Several runs of the simulation are performed using different Courant number which is defined as

$$Cr = \left| u \pm \sqrt{\frac{gA}{b}} \right| \frac{\Delta t}{\Delta x}.$$

The results of the simulation are compared to the solution using Preissmann scheme of the same problem. From the comparison, the conclusions are then drawn.

IX. RESULTS AND CONCLUSION

- Five runs with different Cr numbers were performed, but for clarity, only three of them were compared to Preissmann scheme.
- One of five runs that reflected the correct conservation of mass is plotted more detailed (see Fig 1, 2, 3)
- All the results are given on pp. 12-16

» Conclusions.

Before making conclusions the meaning of Cr with respect to the Hartree fixed grid method needs to be addressed in order to obtain a better understanding how Cr affected the solution.

Physical meaning of Cr on the Hartree scheme is a measure of how far the trajectories of the characteristic from the computational points of the previous time step. If the Cr numbers are somewhat intersect "x-axis" somewhere near computational points, then the results are better compared to the case that the trajectories fall far from computational points.

All the conclusions drawn in the next paragraph refer to Fig. 4 on page 13

Fig.1. Discharge Hydrographs at Several Stations

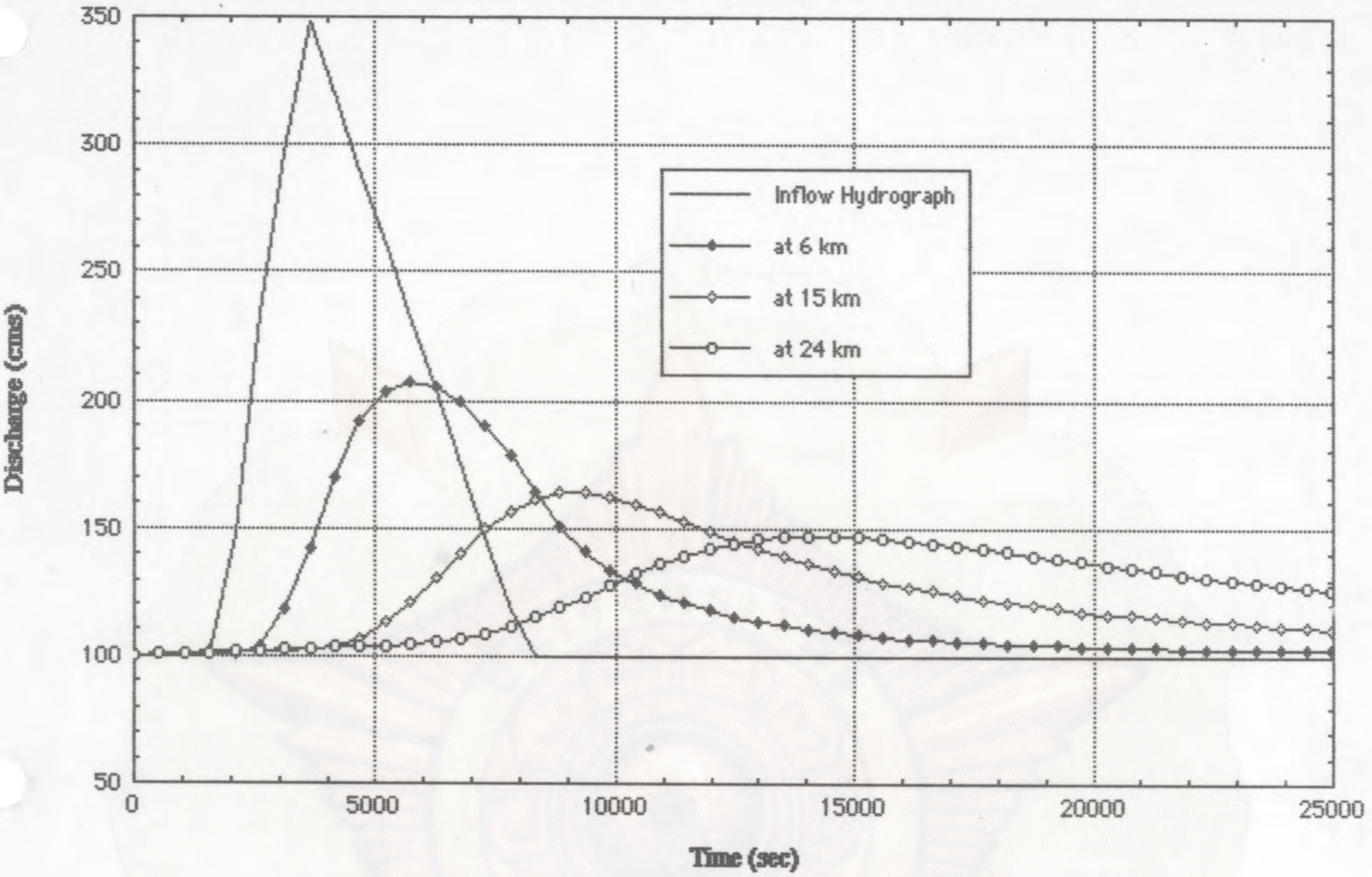
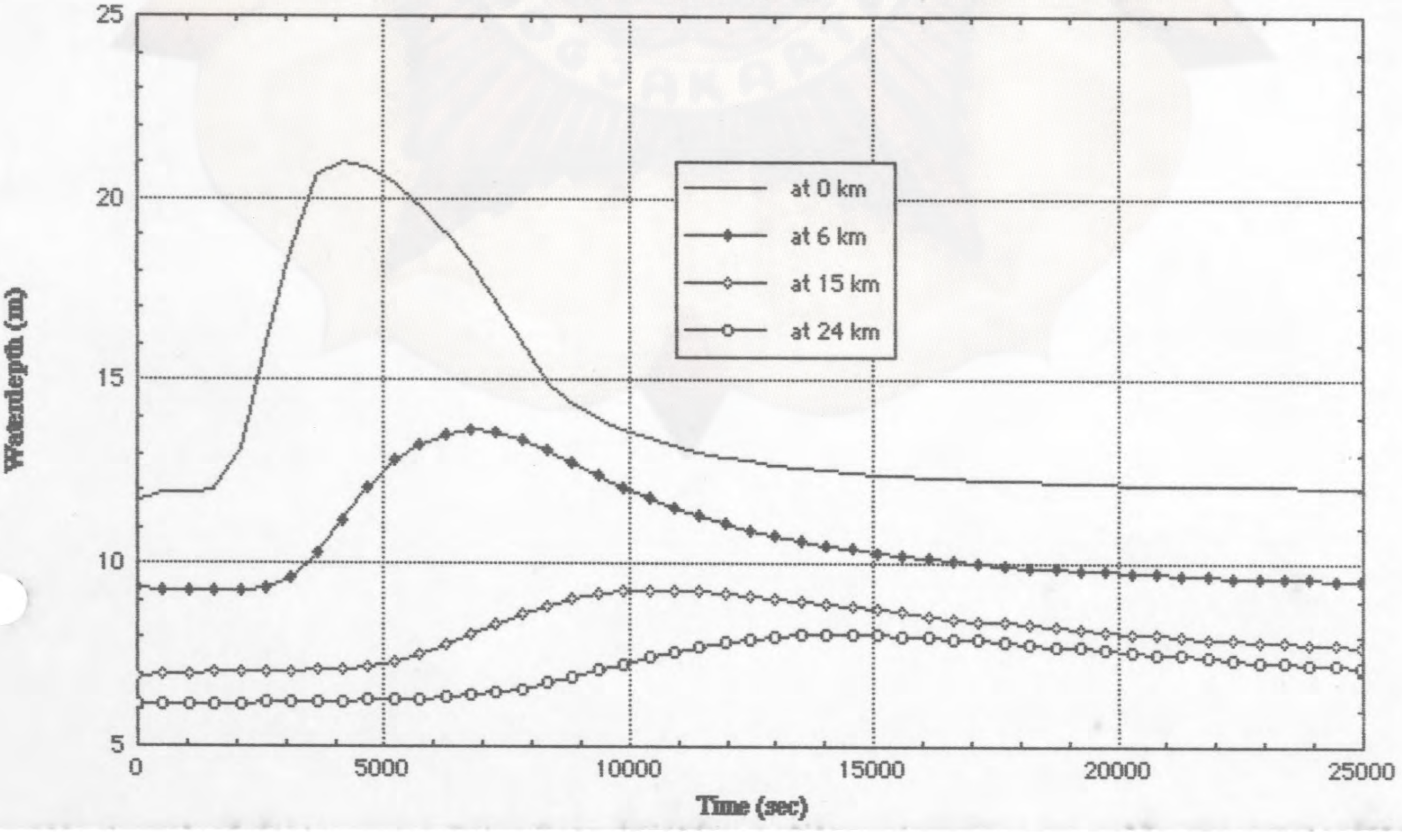


Fig.2. Waterdepth Hydrographs at Several Stations



13/22

Fig.3 Waterdepth Profiles Along The Channel

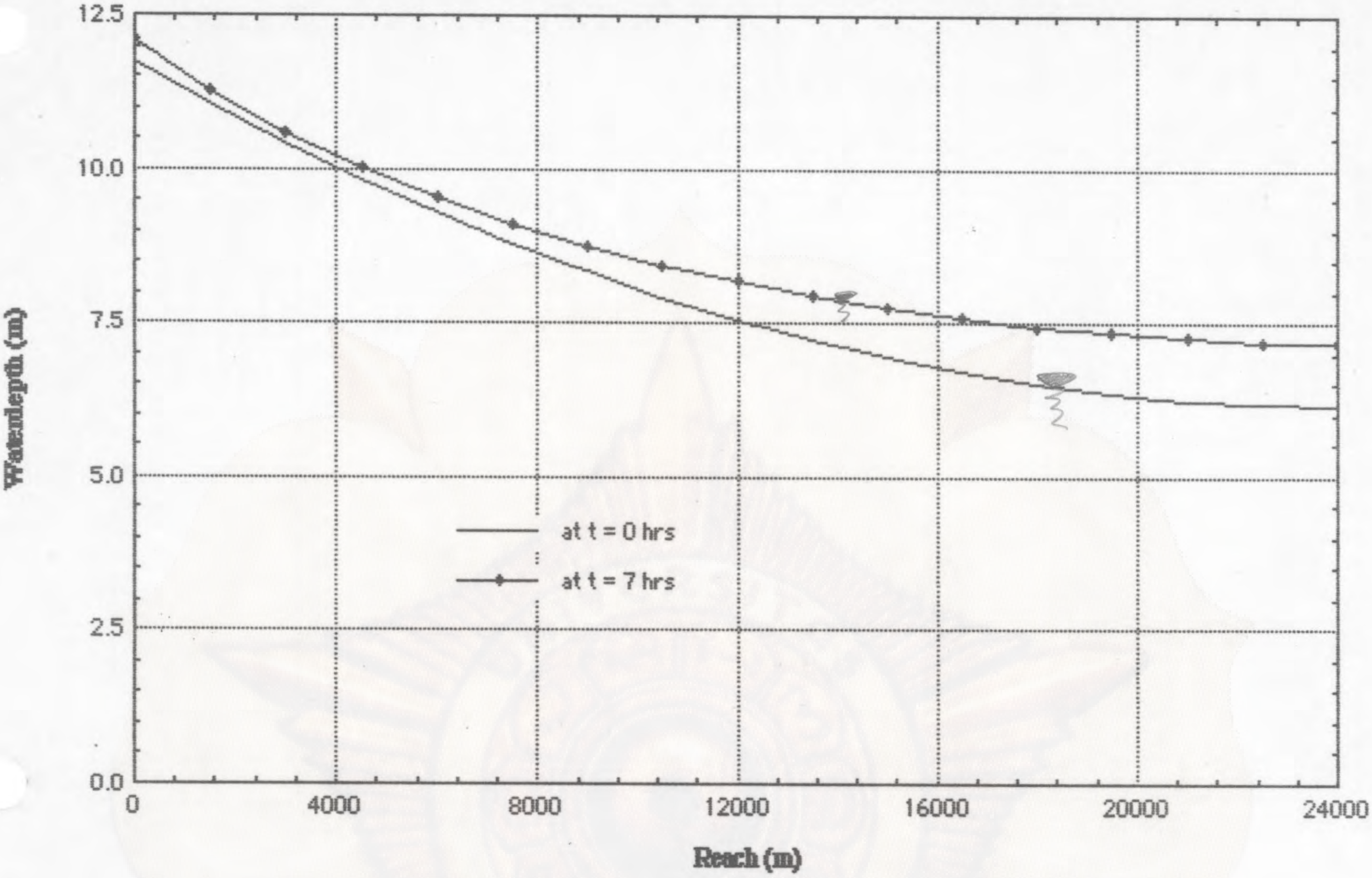
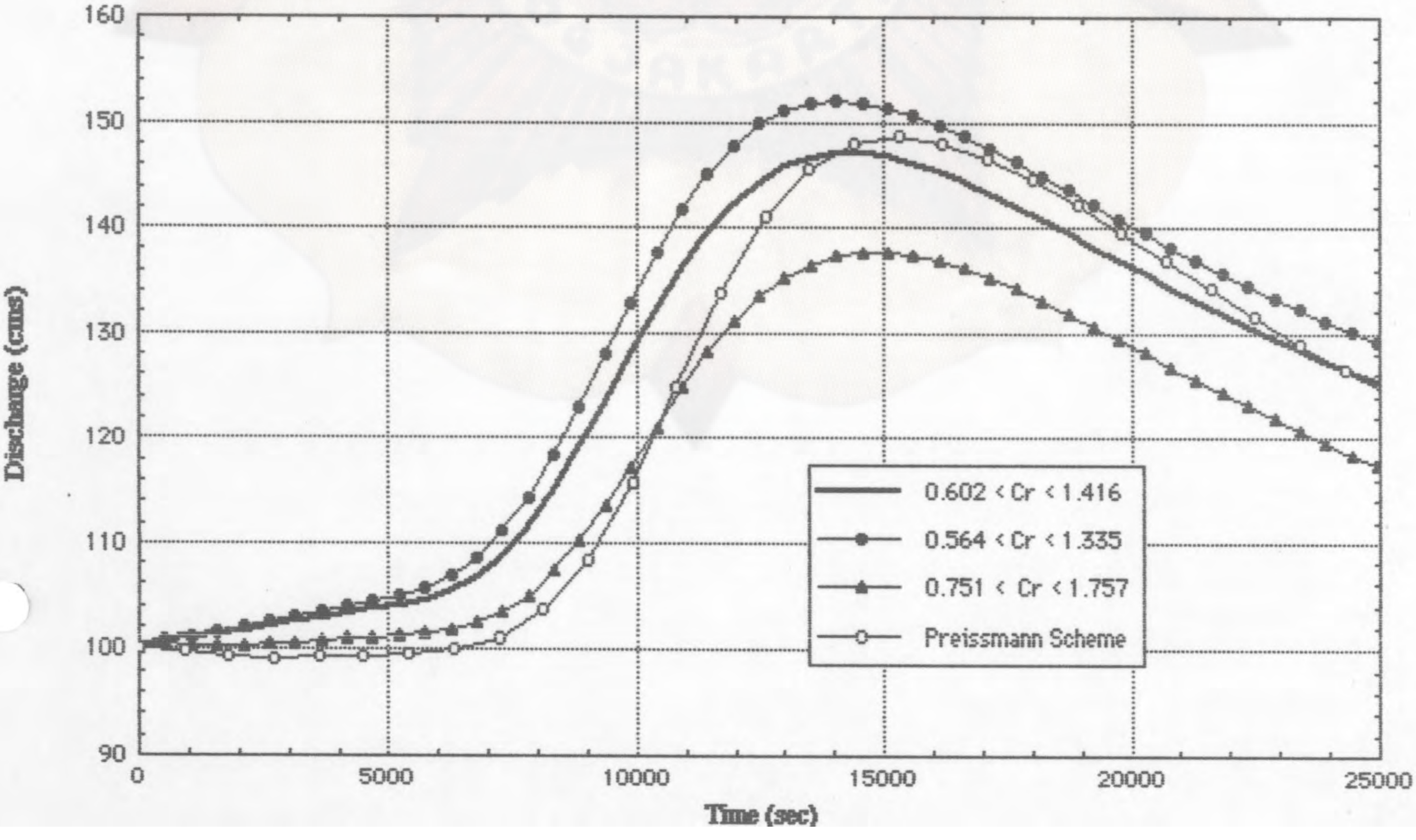
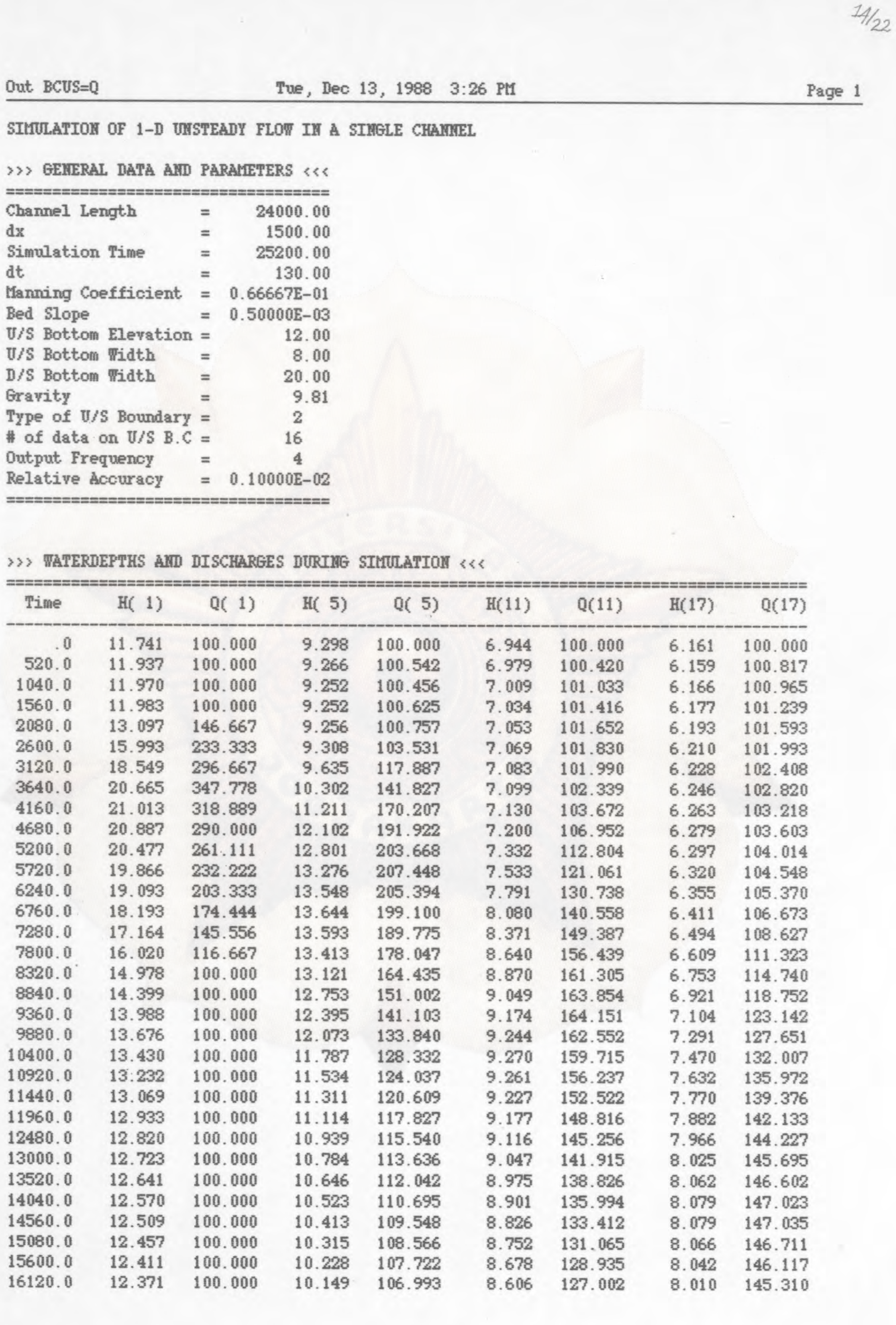


Fig.4. The Hartree vs Preissmann Schemes





14/22

```
Out BCUS=Q                    Tue, Dec 13, 1988  3:26 PM                          Page 1
______________________________________________________________________________________

SIMULATION OF 1-D UNSTEADY FLOW IN A SINGLE CHANNEL

>>> GENERAL DATA AND PARAMETERS <<<
===================================
Channel Length         =    24000.00
dx                     =     1500.00
Simulation Time        =    25200.00
dt                     =      130.00
Manning Coefficient    =  0.66667E-01
Bed Slope              =  0.50000E-03
U/S Bottom Elevation   =       12.00
U/S Bottom Width       =        8.00
D/S Bottom Width       =       20.00
Gravity                =        9.81
Type of U/S Boundary   =        2
# of data on U/S B.C   =       16
Output Frequency       =        4
Relative Accuracy      =  0.10000E-02
===================================


>>> WATERDEPTHS AND DISCHARGES DURING SIMULATION <<<
=========================================================================================
  Time      H( 1)     Q( 1)      H( 5)     Q( 5)       H(11)     Q(11)       H(17)    Q(17)
-----------------------------------------------------------------------------------------
      .0    11.741   100.000     9.298   100.000       6.944   100.000       6.161   100.000
   520.0    11.937   100.000     9.266   100.542       6.979   100.420       6.159   100.817
  1040.0    11.970   100.000     9.252   100.456       7.009   101.033       6.166   100.965
  1560.0    11.983   100.000     9.252   100.625       7.034   101.416       6.177   101.239
  2080.0    13.097   146.667     9.256   100.757       7.053   101.652       6.193   101.593
  2600.0    15.993   233.333     9.308   103.531       7.069   101.830       6.210   101.993
  3120.0    18.549   296.667     9.635   117.887       7.083   101.990       6.228   102.408
  3640.0    20.665   347.778    10.302   141.827       7.099   102.339       6.246   102.820
  4160.0    21.013   318.889    11.211   170.207       7.130   103.672       6.263   103.218
  4680.0    20.887   290.000    12.102   191.922       7.200   106.952       6.279   103.603
  5200.0    20.477   261.111    12.801   203.668       7.332   112.804       6.297   104.014
  5720.0    19.866   232.222    13.276   207.448       7.533   121.061       6.320   104.548
  6240.0    19.093   203.333    13.548   205.394       7.791   130.738       6.355   105.370
  6760.0    18.193   174.444    13.644   199.100       8.080   140.558       6.411   106.673
  7280.0    17.164   145.556    13.593   189.775       8.371   149.387       6.494   108.627
  7800.0    16.020   116.667    13.413   178.047       8.640   156.439       6.609   111.323
  8320.0    14.978   100.000    13.121   164.435       8.870   161.305       6.753   114.740
  8840.0    14.399   100.000    12.753   151.002       9.049   163.854       6.921   118.752
  9360.0    13.988   100.000    12.395   141.103       9.174   164.151       7.104   123.142
  9880.0    13.676   100.000    12.073   133.840       9.244   162.552       7.291   127.651
 10400.0    13.430   100.000    11.787   128.332       9.270   159.715       7.470   132.007
 10920.0    13.232   100.000    11.534   124.037       9.261   156.237       7.632   135.972
 11440.0    13.069   100.000    11.311   120.609       9.227   152.522       7.770   139.376
 11960.0    12.933   100.000    11.114   117.827       9.177   148.816       7.882   142.133
 12480.0    12.820   100.000    10.939   115.540       9.116   145.256       7.966   144.227
 13000.0    12.723   100.000    10.784   113.636       9.047   141.915       8.025   145.695
 13520.0    12.641   100.000    10.646   112.042       8.975   138.826       8.062   146.602
 14040.0    12.570   100.000    10.523   110.695       8.901   135.994       8.079   147.023
 14560.0    12.509   100.000    10.413   109.548       8.826   133.412       8.079   147.035
 15080.0    12.457   100.000    10.315   108.566       8.752   131.065       8.066   146.711
 15600.0    12.411   100.000    10.228   107.722       8.678   128.935       8.042   146.117
 16120.0    12.371   100.000    10.149   106.993       8.606   127.002       8.010   145.310
```

15/22

```
Out BCUS=Q                      Tue, Dec 13, 1988  3:26 PM                          Page 2
16640.0   12.336   100.000   10.079   106.363   8.536   125.245   7.971   144.338
17160.0   12.305   100.000   10.015   105.818   8.469   123.647   7.926   143.243
17680.0   12.278   100.000    9.958   105.343   8.403   122.190   7.879   142.058
18200.0   12.254   100.000    9.907   104.928   8.340   120.861   7.828   140.813
18720.0   12.233   100.000    9.860   104.565   8.280   119.646   7.776   139.530
19240.0   12.214   100.000    9.818   104.246   8.223   118.533   7.723   138.228
19760.0   12.197   100.000    9.780   103.965   8.168   117.512   7.670   136.923
20280.0   12.182   100.000    9.745   103.716   8.115   116.574   7.618   135.626
20800.0   12.169   100.000    9.713   103.495   8.065   115.710   7.566   134.349
21320.0   12.157   100.000    9.684   103.299   8.018   114.914   7.514   133.097
21840.0   12.146   100.000    9.658   103.124   7.974   114.179   7.465   131.878
22360.0   12.136   100.000    9.634   102.968   7.931   113.500   7.416   130.696
22880.0   12.127   100.000    9.612   102.828   7.891   112.872   7.369   129.553
23400.0   12.119   100.000    9.592   102.702   7.853   112.290   7.324   128.452
23920.0   12.112   100.000    9.573   102.589   7.818   111.750   7.280   127.396
24440.0   12.105   100.000    9.556   102.487   7.784   111.250   7.239   126.383
24960.0   12.099   100.000    9.540   102.394   7.752   110.786   7.199   125.415
===================================================================================


 WATER PROFILE AND DISCHARGES ALONG THE CHANNEL
==============================================
 Channel       t = 0.0 sec       t = 25200.0 sec
  Reach     Depth      Disch     Depth      Disch
----------------------------------------------
      .0   11.741    100.000    12.097    100.000
  1500.0   11.065    100.000    11.275     99.976
  3000.0   10.432    100.000    10.598    100.640
  4500.0    9.844    100.000    10.025    101.445
  6000.0    9.298    100.000     9.537    102.372
  7500.0    8.797    100.000     9.117    103.419
  9000.0    8.339    100.000     8.755    104.588
 10500.0    7.925    100.000     8.443    105.888
 12000.0    7.554    100.000     8.174    107.328
 13500.0    7.227    100.000     7.942    108.920
 15000.0    6.944    100.000     7.745    110.675
 16500.0    6.704    100.000     7.578    112.602
 18000.0    6.508    100.000     7.442    114.714
 19500.0    6.356    100.000     7.334    117.019
 21000.0    6.247    100.000     7.254    119.528
 22500.0    6.182    100.000     7.205    122.247
 24000.0    6.161    100.000     7.189    125.180
==============================================


STATISTICS OF THE SIMULATION
===========================
Min. Velocity   =       .812
Max. Velocity   =      2.104
Min. Waterdepth =      6.159
Max. Waterdepth =     21.015
Min. Celerity   =      7.773
Max. Celerity   =     14.358
---------------------------
Min. Courant #  =       .602
Max. Courant #  =      1.416
---------------------------
```

16/22

```
Volume Inflow   = 0.3319E+07
Volume Outflow  = 0.3156E+07
Percentage      =   -4.917 %
===========================
```

1) Solution using method of characteristic is <u>sensitive</u> to Courant number, user must always be aware of it. The solution might become unreasonable if the Courant number is so bad chosen.

2) But the problem on point 1) is very <u>easily</u> be handled by choosing a Cr that gives a <u>correct conservation of mass</u>. ($0.602 \leq Cr \leq 1.416$).

3) For the Cr that gives a correct conservation of mass, the result using method of characteristic has some characteristics as follows:

a). It resembles the solution using Preissmann scheme.

b). The peak discharge comes earlier than Preissmann does, but the tail of the hydrograph decreases slower than Preissmann does.

4) Characteristic approach is much ~~more~~ easier to program, therefore much more appealing compared to Preissmann which need delicate discretization.

6) According to point 3) then the characteristic approach is better for flood warning system since it predicts the peak discharge earlier (on the save side !).
(But this may not be generally true)

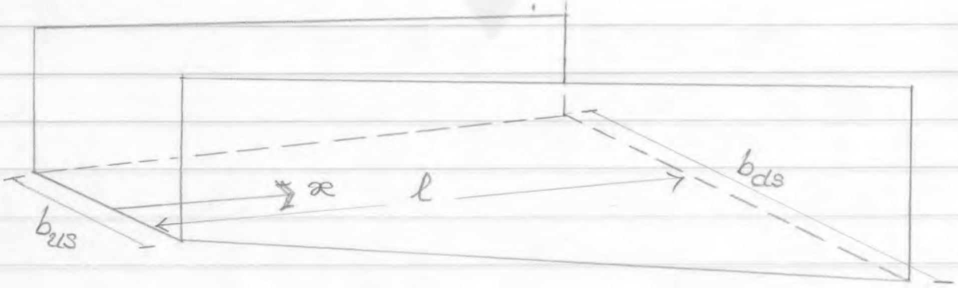
Note: It will be more interesting if the results can be compared to another method of characteristic using a variable grid. Unfortunately one of our program dealing with this grid is stuck due to some errors. From theor~~i~~etical point of view, a variable grid method must give~~s~~ better results than a fixed-grid does, since the only approximation used in a variable-grid is evaluation of integration of the governing eqn.

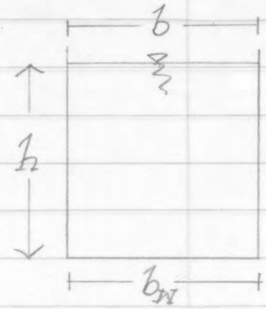
X. REMARKS

It is worth~~y~~ ~~to~~ mentioning that in order to be able to expand the present program for future, all the codes that handle geometries of the channel are written as a seperate subroutines/files. So one can modified the subroutines to suit to his/her needs.

In the next section all the formulas which are used in this subroutines are derived.

X.1. Geometry-correlated formulas.





- $b_N(x) = b_{us} + \frac{x}{\ell}(b_{ds} - b_{us})$. (50)
- $A(b_N, h) = b_N(x) \times h$ (51)
- $P(b_N, h) = b_N + 2h$ (52)
- $b(x) = b_N(x)$ (53)
- $\left(\frac{\partial A}{\partial x}\right)_{h=const} = \frac{h}{\ell}(b_{ds} - b_{us})$ (54)
- $\left(\frac{\partial A}{\partial h}\right) = b$ (55)
- $\frac{\partial}{\partial h}\left(\frac{\partial A}{\partial x}\right)_{h=const} = \frac{b_{ds} - b_{us}}{\ell}$ (56)
- $\frac{\partial P}{\partial h} = 2$ (57)
- $\frac{\partial R}{\partial h} = \frac{\frac{\partial A}{\partial h}P - \frac{\partial P}{\partial h}A}{P^2}$ (58)
- $S_f = \frac{n^2 u^2}{R^{4/3}}$ (59)
- $\frac{\partial S_f}{\partial h} = -\frac{4}{3} n^2 u^2 R^{-7/3} \frac{\partial R}{\partial h}$ (60)
- $c^2 = \frac{gA}{b}$ (61)

$$2c\frac{\partial c}{\partial h} = g\left[\frac{\frac{\partial A}{\partial h}b - \cancelto{0}{\frac{\partial b}{\partial h}}A}{b^2}\right] = g$$

$$\frac{\partial c}{\partial h} = \frac{g}{2c} \qquad (62)$$

- $u = \frac{1}{n} R^{2/3} S^{1/2}$

$$\frac{\partial u}{\partial h} = \frac{2}{3}\frac{1}{n} R^{-\frac{1}{3}} S^{\frac{1}{2}} \frac{\partial R}{\partial h} \qquad (63)$$

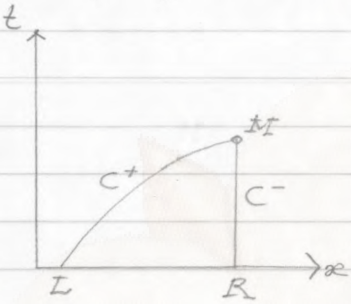
X.2. IMPROVEMENT

Some suggestions to improve the present program are discussed in this section for future use.

a. General Cross-Section

If the term $\frac{\partial b}{\partial h}$ can not be neglected then one has to use "full-blown" working eqns (6 a, b, c). By including this term into the working eqns makes the present program capable to handle arbitrary cross-sections, but the integration evaluation needs delicate programming.

b. The present program is able to handle the following situations, but has not been fully tested !



"critical flow"

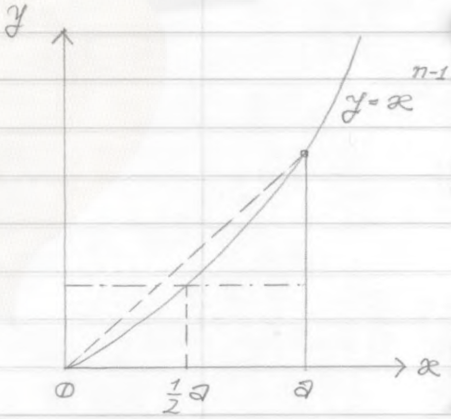
c. Integration Formulas.

In the present program, integration is approximated using trapezoidal rule ie

$$\int_a^b \overset{n}{f}(x)\,dx \approx \frac{1}{2}(b-a)\left[\overset{n}{f}(a)+\overset{n}{f}(b)\right]$$

Let's consider a simple example :

$$E(\text{exact}) = \int_0^a x^{n-1}\,dx = \frac{1}{n}x^n\Big]_0^a = \frac{1}{n}a^n \quad , \text{ let } n \geqslant 2$$



Let use trapezoidal rule to evaluate E :

$$T = \int_0^a x^{n-1}\,dx \approx \frac{1}{2}a\left[a^{n-1}\right] = \frac{1}{2}a^n$$

Let use rectangular rule to evaluate E :

$$R = \int_0^a x^{n-1}\,dx \approx a\left(\frac{a}{2}\right)^{n-1} = \left(\frac{1}{2}\right)^{n-1}a^n$$

From the figure it is obvious that T never has a change to be equal to E, but this is not the case for R.

Now let calculate the differences between two rules from the exact solution :

$$|T-E| = \left|\frac{1}{2}-\frac{1}{n}\right|\left|a^n\right| \quad \text{and} \quad |R-E| = \left|\frac{1}{2^{n-1}}-\frac{1}{n}\right|\left|a^n\right|$$

as $n \to$ large
$$\left.\begin{aligned} |T-E| &\approx \frac{1}{2}\left|a^n\right| \\ |R-E| &\approx \frac{1}{2^{n-1}}\left|a^n\right| \end{aligned}\right\} \therefore R \text{ is better !}$$

For $n = 3$ then $$\left.\begin{aligned} |T-E| &= \tfrac{1}{6}|a|^n \\ |R-E| &= \tfrac{1}{12}|a|^n \end{aligned}\right\} \therefore R \text{ is better !}$$

Consider the case $I = \int_{t_1}^{t_2} S_f \, dt = n^2 \int_{t_1}^{t_2} \left(\frac{u}{R^{2/3}}\right)^2 dt$.

In the present program, the evaluation of I is

$$I \approx n^2 \tfrac{1}{2}(t_2 - t_1)\left[\left(\frac{u}{R^{2/3}}\right)_1^2 + \left(\frac{u}{R^{2/3}}\right)_2^2\right]$$

The accuracy will be improved if I is evaluated as follows :

$$I \approx n^2 (t_2 - t_1)\left[\frac{(u/R^{2/3})_1 + (u/R^{2/3})_2}{2}\right]^2 . \qquad (64).$$

Good analysis.

d. <u>The proposed formulation of the working equations</u>.

Recall : $\sqrt{S_f} \approx \frac{nu}{R^{2/3}}$.

d.1 <u>Upstream Boundary Conditions</u>.

a) $h = h(t)$ given as an u/s b.c :

See Section V.1.a, step 5 :

Eqn. (16) : $$u_R - 2c_R + \tfrac{1}{2}g\Delta t\left(S_o + \frac{uE}{bc}\right)_R + \tfrac{1}{2}g\Delta t\left(S_o + \frac{uE}{bc}\right)_M - g\Delta t\left(\frac{\sqrt{S_{f_R}} + \sqrt{S_{f_M}}}{2}\right)^2 - u_M + 2c_M = 0$$

$$\underbrace{u_R - 2c_R + \tfrac{1}{2}g\Delta t\left(S_o - \tfrac{1}{2}S_f + \frac{uE}{bc}\right)_R}_{GM_R} + \tfrac{1}{2}g\Delta t\left(S_o - \tfrac{1}{2}S_f + \frac{uE}{bc}\right)_M - \tfrac{1}{2}g\Delta t\sqrt{S_{f_R}S_{f_M}} - u_M + 2c_M = 0$$

$$\underbrace{GM_R + 2c_M + \tfrac{1}{2}g\Delta t S_o}_{z_3} + \underbrace{\left[\tfrac{1}{2}g\Delta t\left\{\left(\frac{E}{bc}\right)_M - \frac{n}{R_M^{2/3}}\sqrt{S_{f_R}}\right\} - 1\right]}_{z_2} u_M \underbrace{- \tfrac{1}{4}g\Delta t\frac{n^2}{R_M^{4/3}}}_{z_1} u_M^2 = 0 . \qquad (65)$$

$$\therefore u_M = \frac{-z_2 - \sqrt{z_2^2 - 4z_1z_3}}{2z_1} \qquad (66)$$

b) $Q = Q(t)$ given as an u/s b.c :

Use eqn. (65) ; and substitute $u_M = \frac{Q_M}{A_M}$ where $Q_M = Q(t = t_M)$.

$$\underbrace{GM_R + \tfrac{1}{2}g\Delta t S_o}_{z_3} + 2C_M + \underbrace{\left[\tfrac{1}{2}g\Delta t\left\{\left(\frac{E}{bc}\right)_M - \frac{n\sqrt{S_{f_R}}}{R_M^{2/3}}\right\} - 1\right]Q_M}_{z_2} A_M^{-1} \underbrace{- \tfrac{1}{4}g\Delta t\, n^2 Q_M^2}_{z_1} A_M^{-2} R_M^{-4/3} = 0 \qquad (67)$$

$$\therefore f_1(h_M) = z_1 A_M^{-2} R_M^{-4/3} + z_2 A_M^{-1} + 2C_M + z_3 = 0 \qquad (68)$$

$$\frac{\partial f_1}{\partial h_M} \approx -z_1\left[2A^{-3}R^{-4/3}\frac{\partial A}{\partial h} + \frac{4}{3}A^{-2}R^{-4/3-1}\frac{\partial R}{\partial h}\right]_M + 2\left(\frac{\partial c}{\partial h}\right)_M - z_2 A_M^{-2} + A_M^{-1}\frac{\partial z_2}{\partial h_M}$$

$$\approx -z_1 A^{-2}R^{-4/3}\underbrace{\left[2A^{-1}\frac{\partial A}{\partial h} + \frac{4}{3}R^{-1}\frac{\partial R}{\partial h}\right]_M}_{z_5} + 2\left(\frac{\partial c}{\partial h}\right)_M - \frac{z_2}{A_M^2} +$$

$$A_M^{-1}\left[\underbrace{\frac{1}{2}g\Delta t\, Q_M\left\{\left(\frac{\frac{\partial E}{\partial h}c - \frac{\partial c}{\partial h}E}{bc^2}\right)_M + \frac{2}{3}\frac{n\sqrt{S_{f_R}}}{R_M^{5/3}}\left(\frac{\partial R}{\partial h}\right)_M\right\}}_{z_4}\right] \qquad (69)$$

$$\therefore \frac{\partial f_1}{\partial h_M} \approx \frac{-1}{A_M^2}\left(\frac{z_1 z_5}{R_M^{4/3}} + z_2\right) + \frac{z_4}{A_M} + 2\left(\frac{\partial c}{\partial h}\right)_M \qquad (70)$$

$$\therefore h_M = h_M - \frac{f_1}{f_1'} \qquad (71)$$

d.2. Downstream Boundary Conditions. $(S_{f_M} \approx S_0)$

Modified eqn (38):

$$u_L + 2c_L + \frac{1}{2}g\Delta t\left(S_0 - \frac{uE}{bc}\right)_L + \frac{1}{2}g\Delta t\left(S_0 - \frac{uE}{bc}\right)_M - g\Delta t\left(\frac{\sqrt{S_{f_L}} + \sqrt{S_{f_M}}}{2}\right)^2 - u_M - 2c_M = 0$$

$$\underbrace{\underbrace{u_L + 2c_L + \frac{1}{2}g\Delta t\left(S_0 - \frac{1}{2}S_{f_L} - \frac{uE}{bc}\right)_L}_{GM_L} + \underbrace{\frac{1}{2}g\Delta t\left(S_0 - \frac{1}{2}S_0 - \sqrt{S_0 S_{f_L}}\right)}_{\frac{1}{2}g\Delta t\left(\frac{1}{2}S_0 - \sqrt{S_0 S_{f_L}}\right)}}_{z_2} - \underbrace{\left[\frac{1}{2}g\Delta t\frac{E}{bc} + 1\right]}_{z_1} u_M - 2c_M = 0 \qquad (72)$$

$$\therefore f_2(h_M) \equiv z_2 - z_1 u_M - 2c_M = 0 \qquad (73)$$

$$\frac{\partial f_2}{\partial h_M} = -2\underbrace{\left(\frac{\partial c}{\partial h}\right)_M}_{z_3} - z_1\underbrace{\left(\frac{\partial u}{\partial h}\right)_M}_{z_5} - u_M\underbrace{\frac{1}{2}g\Delta t\left(\frac{\frac{\partial E}{\partial h}c - \frac{\partial c}{\partial h}E}{bc^2}\right)_M}_{z_4} \qquad (74)$$

$$\therefore h_M = h_M - \frac{f_2}{f_2'} \qquad (75)$$

d.3. Interior Points.

Eqn (72): $u_M + 2c_M = GM_L + \frac{1}{2}g\Delta t\left(S_0 - \frac{1}{2}S_f - \frac{uE}{bc}\right)_M - \frac{1}{2}g\Delta t\sqrt{S_{f_L}S_{f_M}}$

Eqn (65): $u_M - 2c_M = GM_R + \frac{1}{2}g\Delta t\left(S_0 - \frac{1}{2}S_f + \frac{uE}{bc}\right)_M - \frac{1}{2}g\Delta t\sqrt{S_{f_R}S_{f_M}}$ $\pm$

$$2u_M = GM_L + GM_R + \frac{1}{2}g\Delta t\left(2S_0 - S_{f_M}\right) - \frac{1}{2}g\Delta t\left(\sqrt{S_{f_L}} + \sqrt{S_{f_R}}\right)\sqrt{S_{f_M}}$$

$$u_M = \underbrace{\frac{1}{2}\left(GM_L + GM_R\right) + \frac{1}{2}g\Delta t\, S_0}_{z_3}\underbrace{-\frac{1}{4}g\Delta t}_{z_1'} S_{f_M}\underbrace{-\frac{1}{4}g\Delta t\left(\sqrt{S_{f_L}} + \sqrt{S_{f_R}}\right)}_{z_2'}\left(S_{f_M}\right)^{1/2}$$

$$\therefore\ u_M = z_3 + \underbrace{z_1' n^2}_{z_1} R_M^{-4/3} u_M^2 + \underbrace{z_2' n}_{z_2} R_M^{-2/3} u_M$$

$$\therefore\ f_1(h_M, u_M) = z_3 + z_1 R_M^{-4/3} |u_M| u_M + (z_2 R_M^{-2/3} - 1) u_M \qquad (76)$$

$$4c_M = GM_L - GM_R - g\Delta t \left(\frac{uE}{bc}\right)_M - \frac{1}{2} g\Delta t \left(\sqrt{S_{f_L}} - \sqrt{S_{f_R}}\right)\left(S_{f_M}\right)^{1/2}$$

$$c_M = \underbrace{\frac{1}{4}(GM_L - GM_R)}_{z_4} - \underbrace{\frac{1}{4} g\Delta t}_{z_5} \left(\frac{E}{bc}\right)_M u_M - \underbrace{\frac{1}{8} g\Delta t\, n \left(\sqrt{S_{f_L}} - \sqrt{S_{f_R}}\right)}_{z_6} u_M R_M^{-2/3}$$

$$\therefore\ f_2(h_M, u_M) = z_4 + z_5 \left(\frac{E}{bc}\right)_M u_M + z_6 u_M R_M^{-2/3} - c_M \qquad (77)$$

So

$$\swarrow |u_M| u_M$$

$$a_{11} = \frac{\partial f_1}{\partial h_M} = -\frac{4}{3} z_1 R_M^{-(\frac{4}{3}+1)} u_M^2 \left(\frac{\partial R}{\partial h}\right)_M - \frac{2}{3} z_2 u_M R_M^{-(\frac{2}{3}+1)} \left(\frac{\partial R}{\partial h}\right)_M$$

$$= -\frac{2}{3} u_M R_M^{-(\frac{4}{3}+1)} \left(\frac{\partial R}{\partial h}\right)_M \left[2 z_1 |u_M| + z_2 R_M^{\frac{2}{3}} \right] \qquad (78)$$

$$a_{12} = \frac{\partial f_1}{\partial u_M} = 2 z_1 R_M^{-4/3} |u_M| + z_2 R_M^{-2/3} - 1.0 \qquad (79)$$

$$a_{21} = \frac{\partial f_2}{\partial h_M} = z_5 u_M \left(\frac{\frac{\partial E}{\partial h} c - \frac{\partial c}{\partial h} E}{bc^2} \right)_M - \frac{2}{3} z_6 u_M R_M^{-(\frac{2}{3}+1)} \left(\frac{\partial R}{\partial h}\right)_M - \left(\frac{\partial c}{\partial h}\right)_M \qquad (80)$$

$$a_{22} = \frac{\partial f_2}{\partial u_M} = z_5 \left(\frac{E}{bc}\right)_M + z_6 R_M^{-2/3} \qquad (81)$$

Then use eqns. (47), (48), and (49) to solve for Δh_M and Δu_M.

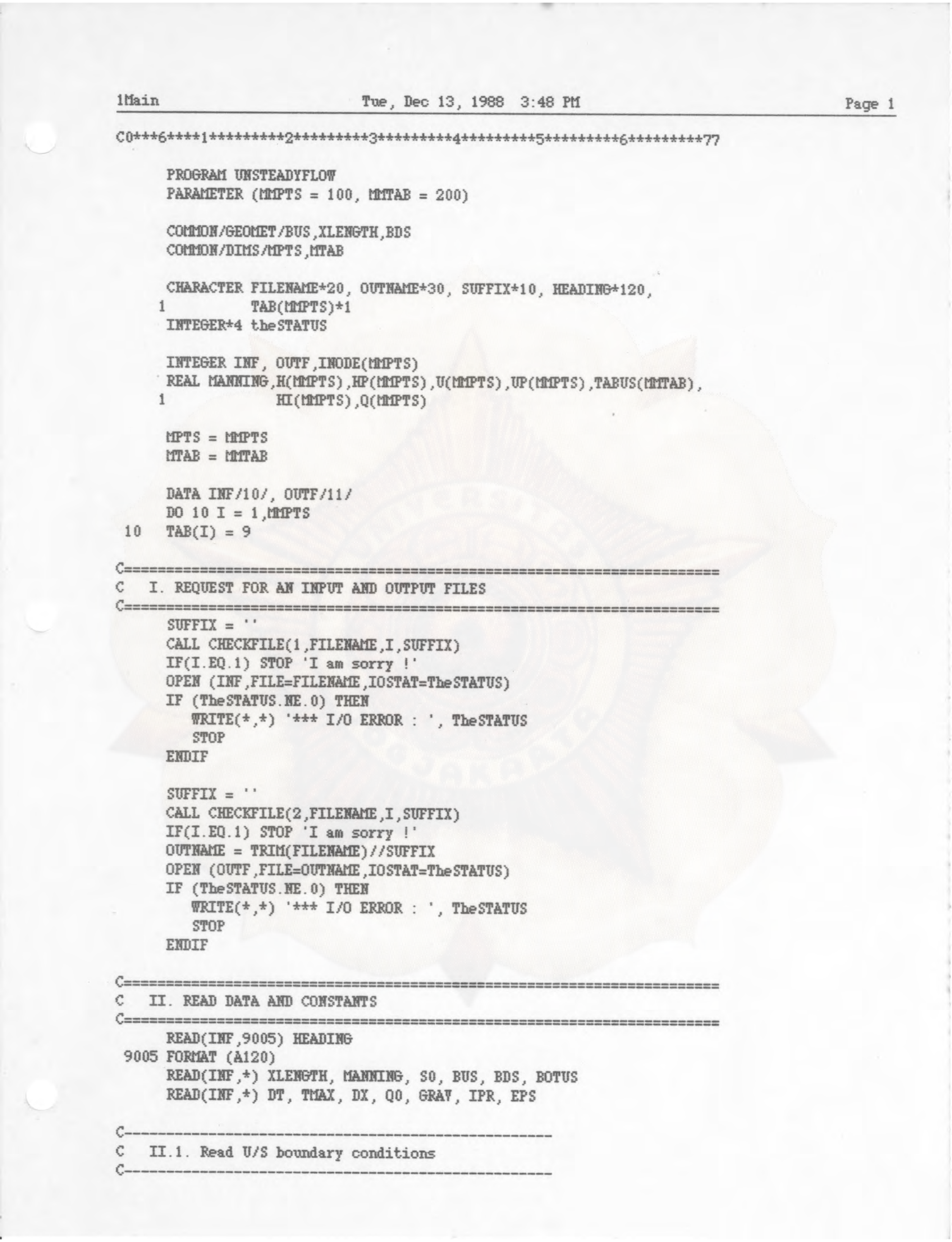
The implementation of this integration formulas into the present program is straight forward and needs only minor changes in the program.

Good in-depth effort.

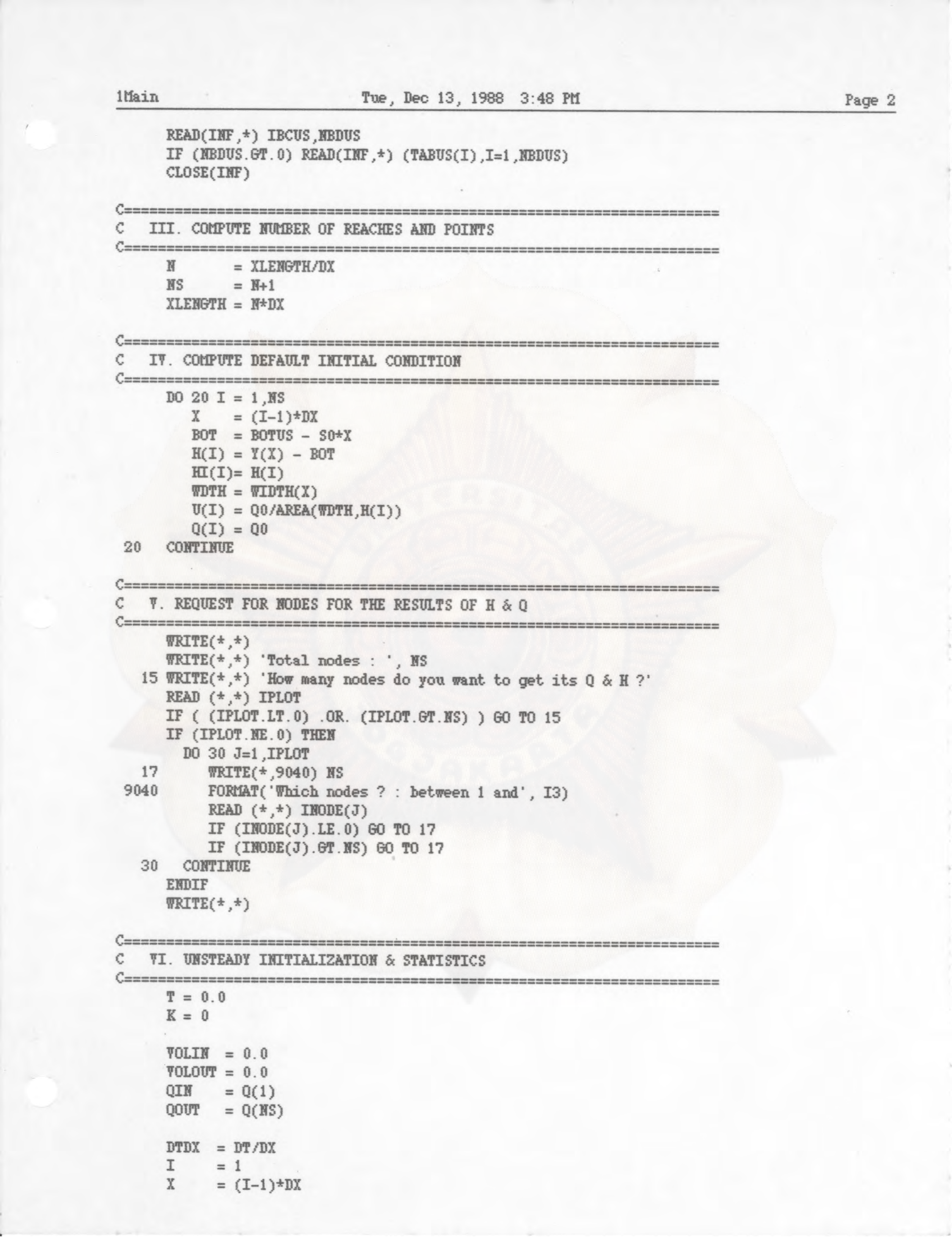
Note :
- The previous integration formula is written in file : 4Boundaries.
- The improved integration formula is written in file : 4ImproveBC

INPUT DATA
and
PROGRAM LISTING
for
THE HARTREE FIXED-GRID METHOD

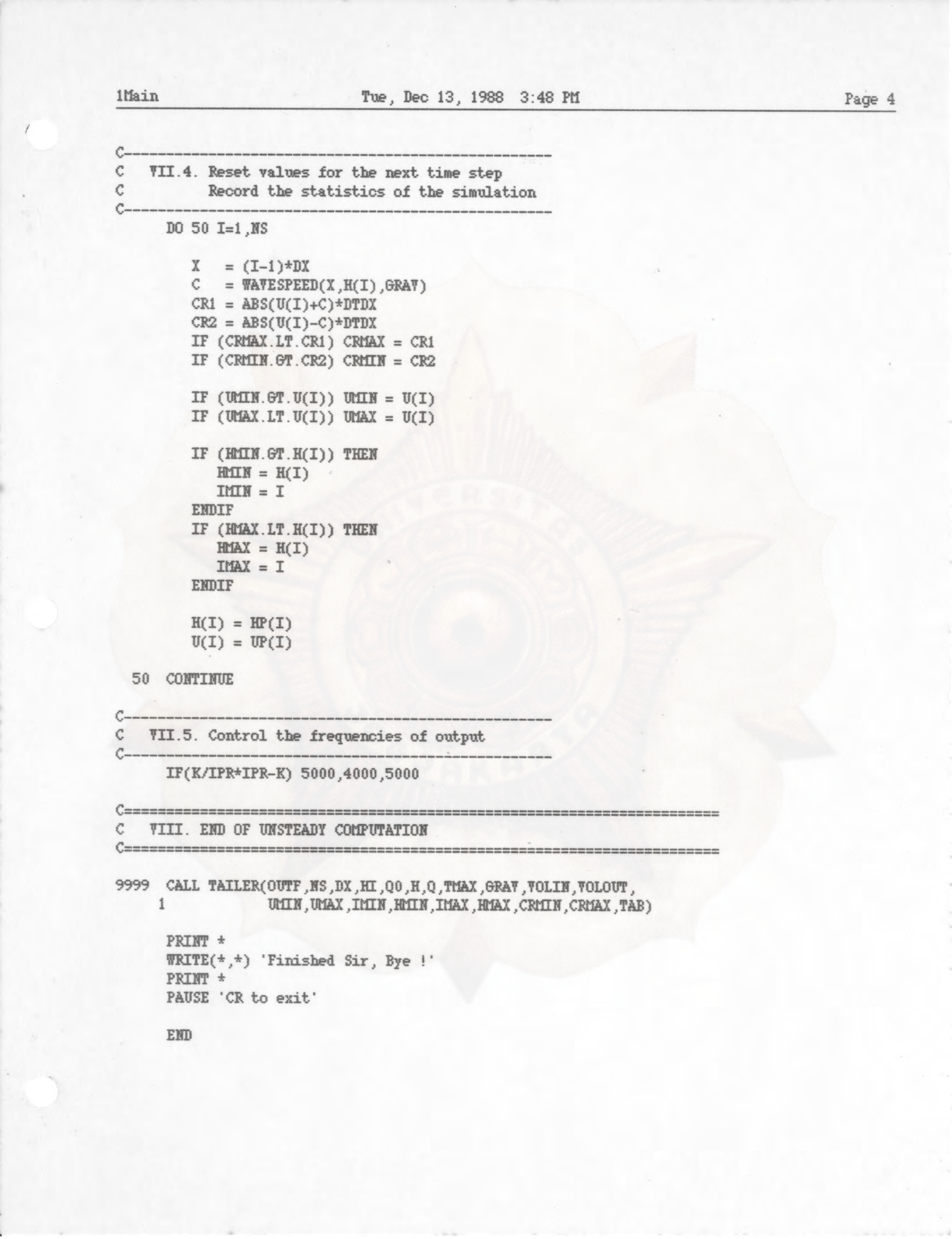
```
SIMULATION OF 1-D UNSTEADY FLOW IN A SINGLE CHANNEL
24000.0  0.0666666666667  0.0005  8.0  20.0 12.0
  130.0  25200.0  1500.0  100.0  9.81  4  1.0E-3
    2        16
 1800.0     100.0
 2700.0     250.0
 3600.0     350.0
 4500.0     300.0
 5400.0     250.0
 6300.0     200.0
 7200.0     150.0
 8100.0     100.0
```



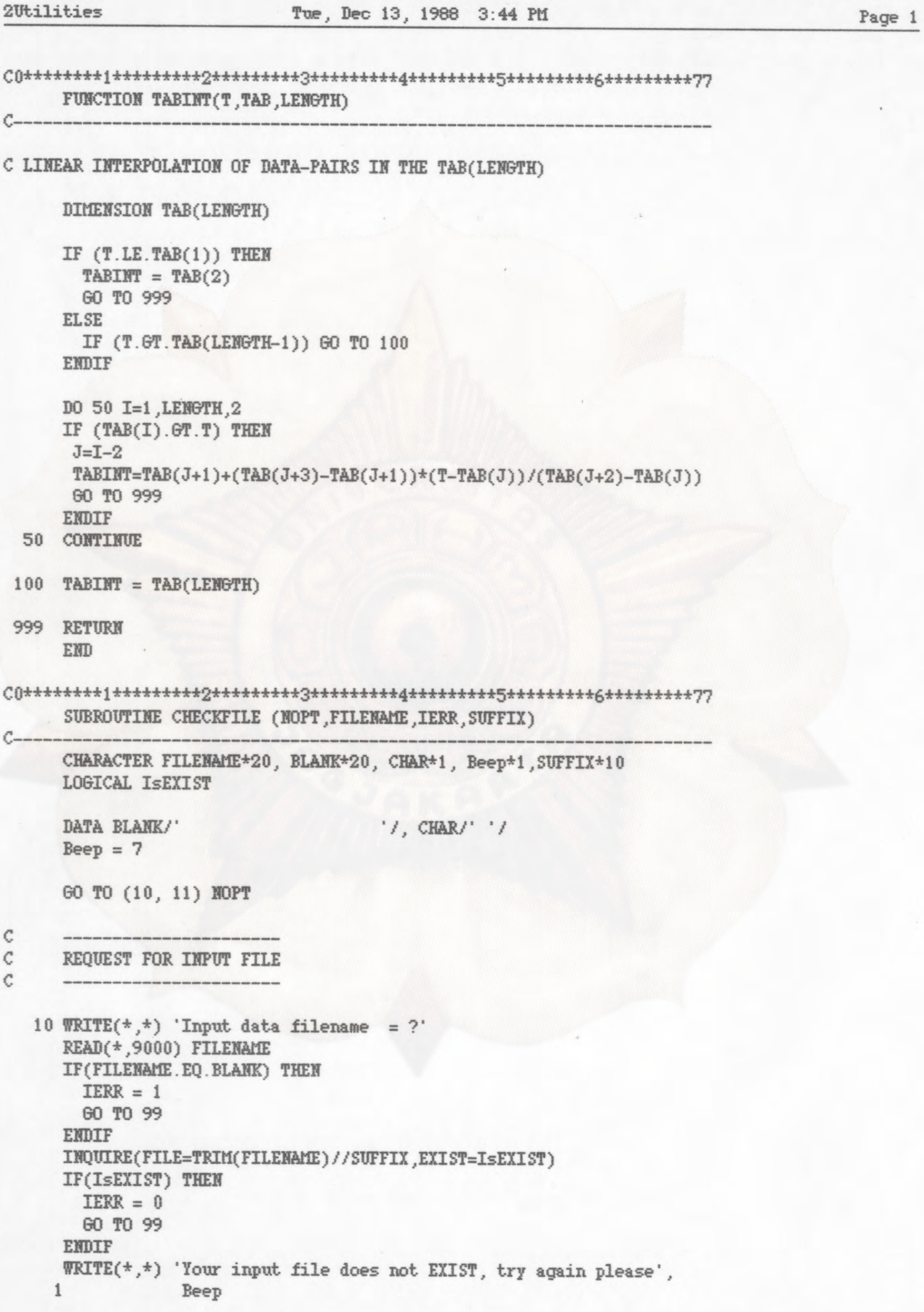
```
C0***6****1*********2*********3*********4*********5*********6*********77

      PROGRAM UNSTEADYFLOW
      PARAMETER (MMPTS = 100, MMTAB = 200)

      COMMON/GEOMET/BUS,XLENGTH,BDS
      COMMON/DIMS/MPTS,MTAB

      CHARACTER FILENAME*20, OUTNAME*30, SUFFIX*10, HEADING*120,
     1          TAB(MMPTS)*1
      INTEGER*4 theSTATUS

      INTEGER INF, OUTF,INODE(MMPTS)
      REAL MANNING,H(MMPTS),HP(MMPTS),U(MMPTS),UP(MMPTS),TABUS(MMTAB),
     1             HI(MMPTS),Q(MMPTS)

      MPTS = MMPTS
      MTAB = MMTAB

      DATA INF/10/, OUTF/11/
      DO 10 I = 1,MMPTS
 10   TAB(I) = 9

C=====================================================================
C   I. REQUEST FOR AN INPUT AND OUTPUT FILES
C=====================================================================
      SUFFIX = ''
      CALL CHECKFILE(1,FILENAME,I,SUFFIX)
      IF(I.EQ.1) STOP 'I am sorry !'
      OPEN (INF,FILE=FILENAME,IOSTAT=TheSTATUS)
      IF (TheSTATUS.NE.0) THEN
         WRITE(*,*) '*** I/O ERROR : ', TheSTATUS
         STOP
      ENDIF

      SUFFIX = ''
      CALL CHECKFILE(2,FILENAME,I,SUFFIX)
      IF(I.EQ.1) STOP 'I am sorry !'
      OUTNAME = TRIM(FILENAME)//SUFFIX
      OPEN (OUTF,FILE=OUTNAME,IOSTAT=TheSTATUS)
      IF (TheSTATUS.NE.0) THEN
         WRITE(*,*) '*** I/O ERROR : ', TheSTATUS
         STOP
      ENDIF

C=====================================================================
C   II. READ DATA AND CONSTANTS
C=====================================================================
      READ(INF,9005) HEADING
 9005 FORMAT (A120)
      READ(INF,*) XLENGTH, MANNING, S0, BUS, BDS, BOTUS
      READ(INF,*) DT, TMAX, DX, Q0, GRAV, IPR, EPS

C--------------------------------------------------
C   II.1. Read U/S boundary conditions
C--------------------------------------------------
```



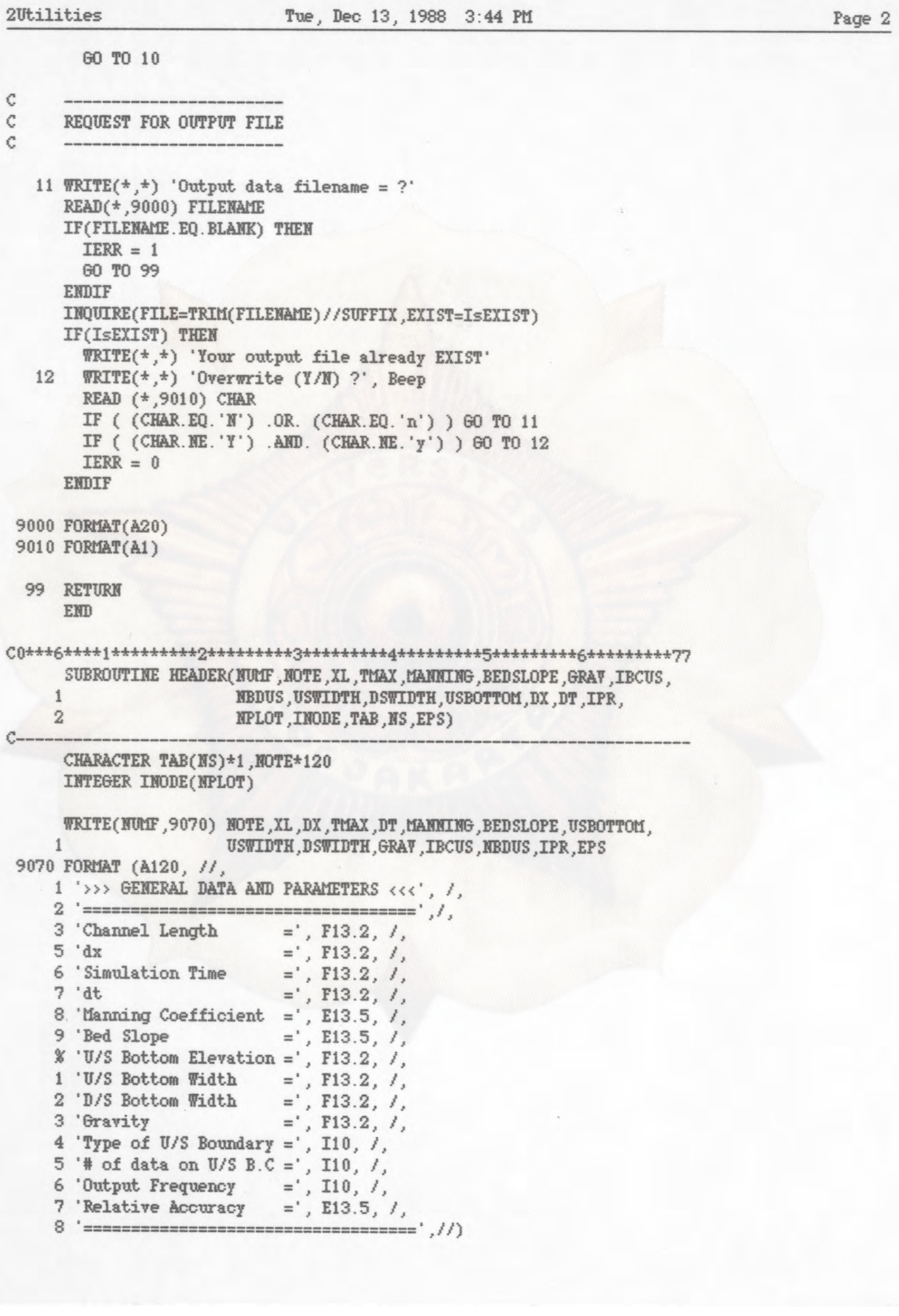
```
      READ(INF,*) IBCUS,NBDUS
      IF (NBDUS.GT.0) READ(INF,*) (TABUS(I),I=1,NBDUS)
      CLOSE(INF)

C========================================================================
C   III. COMPUTE NUMBER OF REACHES AND POINTS
C========================================================================
      N       = XLENGTH/DX
      NS      = N+1
      XLENGTH = N*DX

C========================================================================
C   IV. COMPUTE DEFAULT INITIAL CONDITION
C========================================================================
      DO 20 I = 1,NS
         X    = (I-1)*DX
         BOT  = BOTUS - S0*X
         H(I) = Y(X) - BOT
         HI(I)= H(I)
         WDTH = WIDTH(X)
         U(I) = Q0/AREA(WDTH,H(I))
         Q(I) = Q0
 20   CONTINUE

C========================================================================
C   V. REQUEST FOR NODES FOR THE RESULTS OF H & Q
C========================================================================
      WRITE(*,*)
      WRITE(*,*) 'Total nodes : ', NS
   15 WRITE(*,*) 'How many nodes do you want to get its Q & H ?'
      READ (*,*) IPLOT
      IF ( (IPLOT.LT.0) .OR. (IPLOT.GT.NS) ) GO TO 15
      IF (IPLOT.NE.0) THEN
        DO 30 J=1,IPLOT
   17     WRITE(*,9040) NS
 9040     FORMAT('Which nodes ? : between 1 and', I3)
          READ (*,*) INODE(J)
          IF (INODE(J).LE.0) GO TO 17
          IF (INODE(J).GT.NS) GO TO 17
   30   CONTINUE
      ENDIF
      WRITE(*,*)

C========================================================================
C   VI. UNSTEADY INITIALIZATION & STATISTICS
C========================================================================
      T = 0.0
      K = 0

      VOLIN  = 0.0
      VOLOUT = 0.0
      QIN    = Q(1)
      QOUT   = Q(NS)

      DTDX  = DT/DX
      I     = 1
      X     = (I-1)*DX
      UMIN  = U(I)
      UMAX  = UMIN
      HMIN  = H(I)
      HMAX  = HMIN
      C     = WAVESPEED(X,H(I),GRAV)
      CRMIN = ABS(U(I)-C)*DTDX
      CRMAX = ABS(U(I)+C)*DTDX

      CALL HEADER(OUTF,HEADING,XLENGTH,TMAX,MANNING,S0,GRAV,IBCUS,NBDUS,
     1            BUS,BDS,BOTUS,DX,DT,IPR,IPLOT,INODE,TAB,NS,EPS)

4000  CALL LISTER(OUTF,T,H,Q,NS,IPLOT,INODE,TAB)

      WRITE(*,9100) T,H(1),Q(1),NS,H(NS),U(NS),Q(NS)
9100  FORMAT ('Time: ',F8.1,'  H,Q( 1) = ',F7.2,F9.2,
     %                      '  H,U,Q(',I2,') = ',2F7.2,F9.2)

C=======================================================================
C   VII. BEGINNING OF UNSTEADY COMPUTATION
C=======================================================================
5000  T = T + DT
      IF(T.GT.TMAX) GO TO 9999
      K = K + 1

C--------------------------------------------------
C   VII.1. Upstream Boundary Condition
C--------------------------------------------------
      CALL BCUS(T,IBCUS,TABUS,NBDUS,H(NS),U(NS),H(NS),U(NS),H,U,NS,
     1          GRAV,DT,DX,Manning,S0,UP(1),CM,HP(1),EPS)

      X     = 0.0
      Q(1)  = DISCH(UP(1),X,HP(1))
      VOLIN = VOLIN + 0.5*DT*(QIN+Q(1))
      QIN   = Q(1)

C--------------------------------------------------
C   VII.2. Downstream Boundary Condition
C--------------------------------------------------
      CALL BCDS(T,H(1),U(1),HP(1),UP(1),H,U,NS,
     1          GRAV,DT,DX,Manning,S0,UP(NS),CM,HP(NS),EPS)

      X      = (NS-1)*DX
      Q(NS)  = DISCH(UP(NS),X,HP(NS))
      VOLOUT = VOLOUT + 0.5*DT*(QOUT+Q(NS))
      QOUT   = Q(NS)

C--------------------------------------------------
C   VII.3. Interior Points
C--------------------------------------------------
      DO 40 I=2,N

         CALL CALCM(H(1),U(1),HP(1),UP(1),H(NS),U(NS),HP(NS),UP(NS),
     1              T,H,U,NS,I,GRAV,DT,DX,Manning,S0,UP(I),CM,HP(I),EPS)
         X     = (I-1)*DX
         Q(I)  = DISCH(UP(I),X,HP(I))

   40 CONTINUE
```



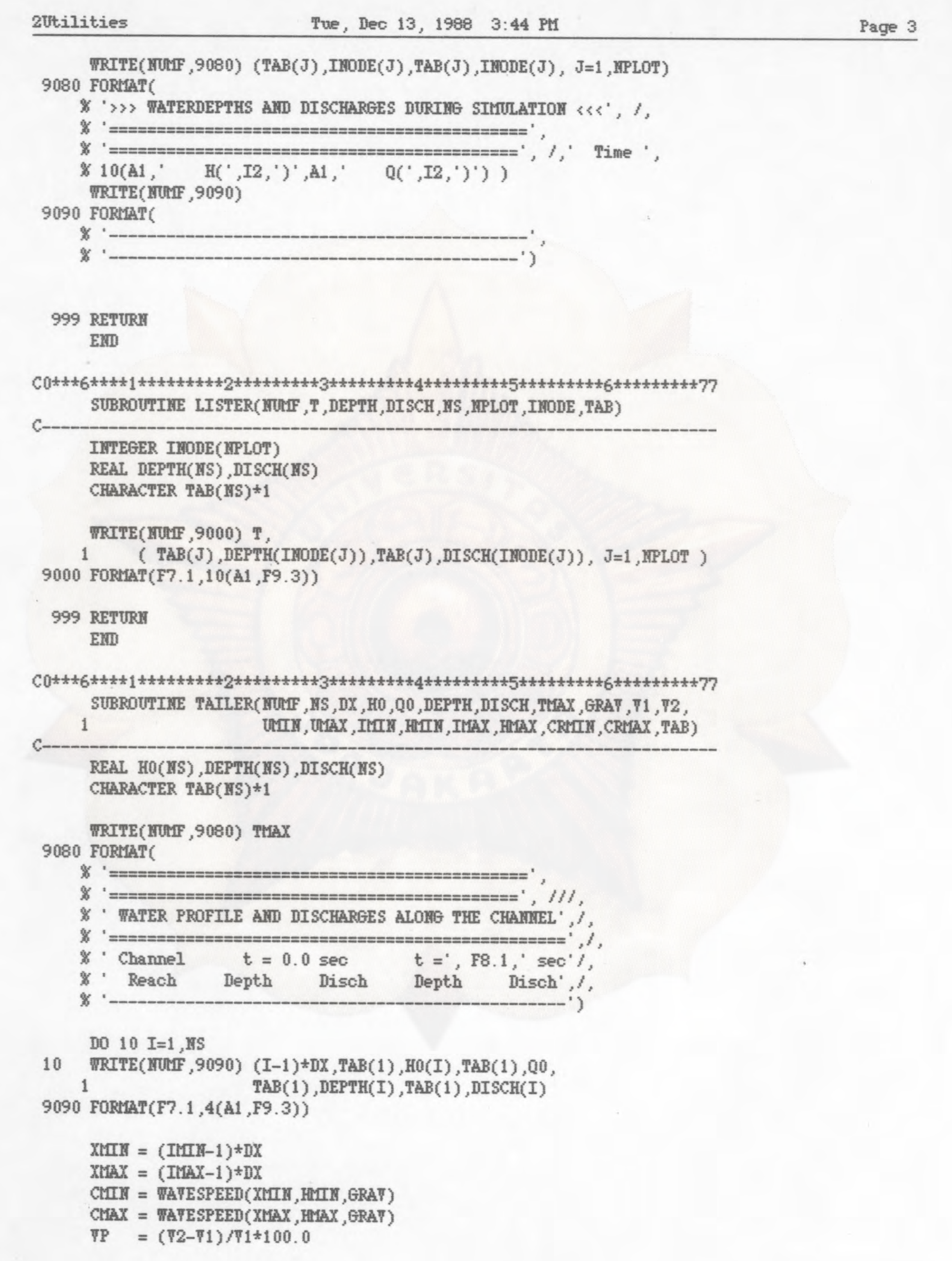
```
C--------------------------------------------------
C   VII.4. Reset values for the next time step
C          Record the statistics of the simulation
C--------------------------------------------------
      DO 50 I=1,NS

         X   = (I-1)*DX
         C   = WAVESPEED(X,H(I),GRAV)
         CR1 = ABS(U(I)+C)*DTDX
         CR2 = ABS(U(I)-C)*DTDX
         IF (CRMAX.LT.CR1) CRMAX = CR1
         IF (CRMIN.GT.CR2) CRMIN = CR2

         IF (UMIN.GT.U(I)) UMIN = U(I)
         IF (UMAX.LT.U(I)) UMAX = U(I)

         IF (HMIN.GT.H(I)) THEN
            HMIN = H(I)
            IMIN = I
         ENDIF
         IF (HMAX.LT.H(I)) THEN
            HMAX = H(I)
            IMAX = I
         ENDIF

         H(I) = HP(I)
         U(I) = UP(I)

  50  CONTINUE

C--------------------------------------------------
C   VII.5. Control the frequencies of output
C--------------------------------------------------
      IF(K/IPR*IPR-K) 5000,4000,5000

C========================================================================
C   VIII. END OF UNSTEADY COMPUTATION
C========================================================================

9999  CALL TAILER(OUTF,NS,DX,HI,Q0,H,Q,TMAX,GRAV,VOLIN,VOLOUT,
     1             UMIN,UMAX,IMIN,HMIN,IMAX,HMAX,CRMIN,CRMAX,TAB)

      PRINT *
      WRITE(*,*) 'Finished Sir, Bye !'
      PRINT *
      PAUSE 'CR to exit'

      END
```



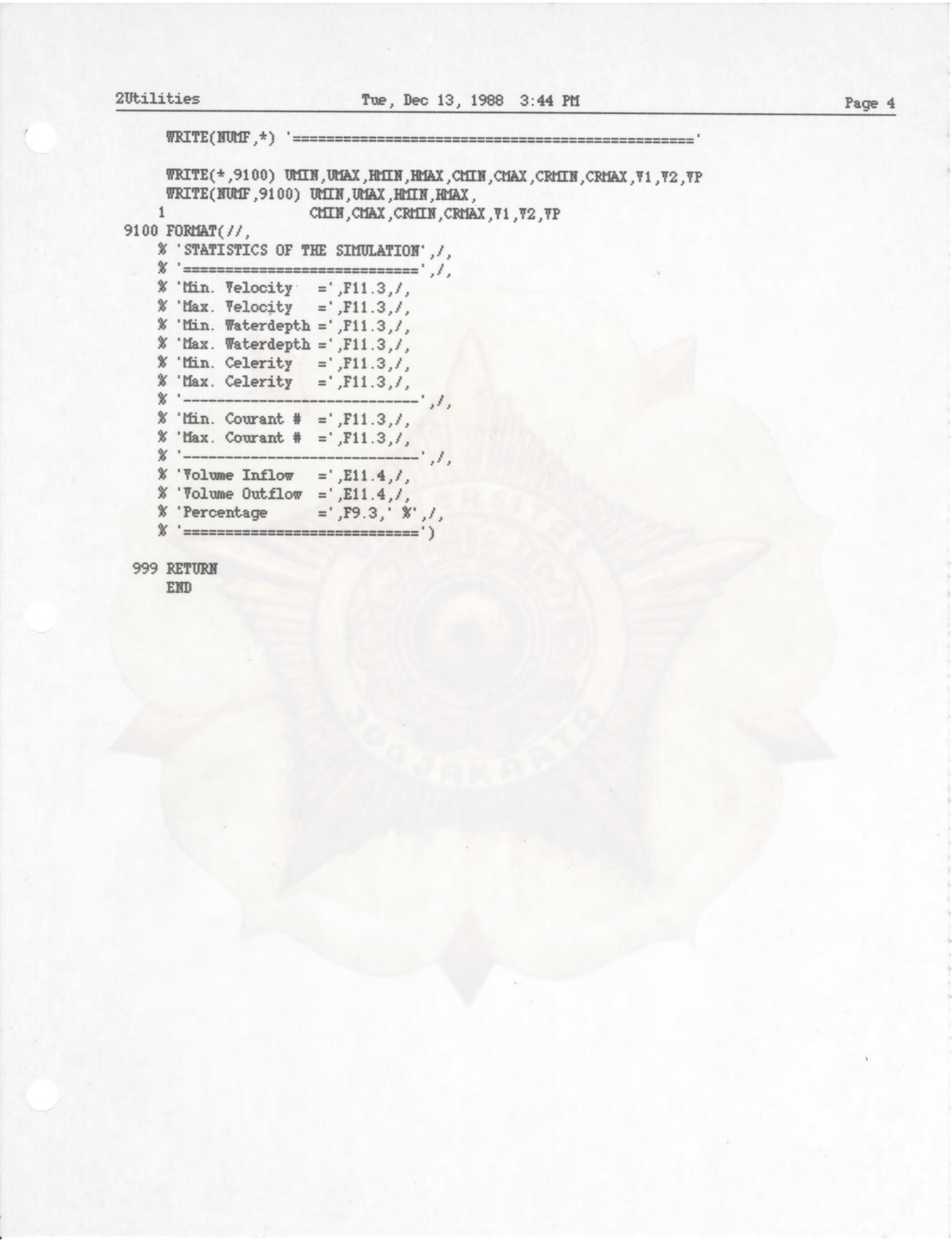
```
C0********1*********2*********3*********4*********5*********6*********77
      FUNCTION TABINT(T,TAB,LENGTH)
C-----------------------------------------------------------------------

C LINEAR INTERPOLATION OF DATA-PAIRS IN THE TAB(LENGTH)

      DIMENSION TAB(LENGTH)

      IF (T.LE.TAB(1)) THEN
        TABINT = TAB(2)
        GO TO 999
      ELSE
        IF (T.GT.TAB(LENGTH-1)) GO TO 100
      ENDIF

      DO 50 I=1,LENGTH,2
      IF (TAB(I).GT.T) THEN
       J=I-2
       TABINT=TAB(J+1)+(TAB(J+3)-TAB(J+1))*(T-TAB(J))/(TAB(J+2)-TAB(J))
       GO TO 999
      ENDIF
   50 CONTINUE

  100 TABINT = TAB(LENGTH)

  999 RETURN
      END

C0********1*********2*********3*********4*********5*********6*********77
      SUBROUTINE CHECKFILE (NOPT,FILENAME,IERR,SUFFIX)
C-----------------------------------------------------------------------
      CHARACTER FILENAME*20, BLANK*20, CHAR*1, Beep*1,SUFFIX*10
      LOGICAL IsEXIST

      DATA BLANK/'                    '/, CHAR/' '/
      Beep = 7

      GO TO (10, 11) NOPT

C     ----------------------
C     REQUEST FOR INPUT FILE
C     ----------------------

   10 WRITE(*,*) 'Input data filename  = ?'
      READ(*,9000) FILENAME
      IF(FILENAME.EQ.BLANK) THEN
        IERR = 1
        GO TO 99
      ENDIF
      INQUIRE(FILE=TRIM(FILENAME)//SUFFIX,EXIST=IsEXIST)
      IF(IsEXIST) THEN
        IERR = 0
        GO TO 99
      ENDIF
      WRITE(*,*) 'Your input file does not EXIST, try again please',
     1            Beep
```



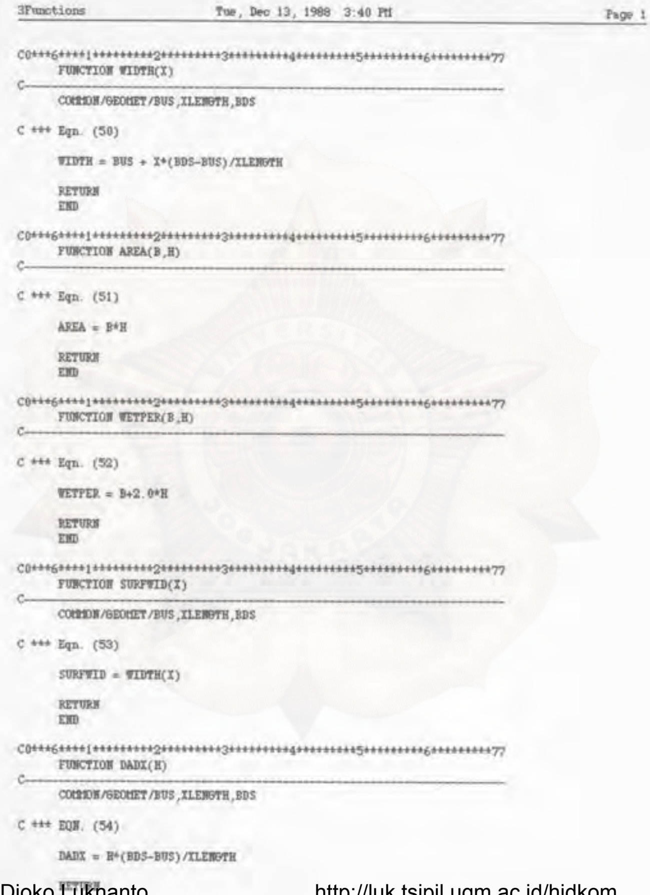
```
        GO TO 10

C     ------------------------
C     REQUEST FOR OUTPUT FILE
C     ------------------------

   11 WRITE(*,*) 'Output data filename = ?'
      READ(*,9000) FILENAME
      IF(FILENAME.EQ.BLANK) THEN
        IERR = 1
        GO TO 99
      ENDIF
      INQUIRE(FILE=TRIM(FILENAME)//SUFFIX,EXIST=IsEXIST)
      IF(IsEXIST) THEN
        WRITE(*,*) 'Your output file already EXIST'
   12   WRITE(*,*) 'Overwrite (Y/N) ?', Beep
        READ (*,9010) CHAR
        IF ( (CHAR.EQ.'N') .OR. (CHAR.EQ.'n') ) GO TO 11
        IF ( (CHAR.NE.'Y') .AND. (CHAR.NE.'y') ) GO TO 12
        IERR = 0
      ENDIF

 9000 FORMAT(A20)
 9010 FORMAT(A1)

   99 RETURN
      END

C0***6****1*********2*********3*********4*********5*********6*********77
      SUBROUTINE HEADER(NUMF,NOTE,XL,TMAX,MANNING,BEDSLOPE,GRAV,IBCUS,
     1                  NBDUS,USWIDTH,DSWIDTH,USBOTTOM,DX,DT,IPR,
     2                  NPLOT,INODE,TAB,NS,EPS)
C-----------------------------------------------------------------------
      CHARACTER TAB(NS)*1,NOTE*120
      INTEGER INODE(NPLOT)

      WRITE(NUMF,9070) NOTE,XL,DX,TMAX,DT,MANNING,BEDSLOPE,USBOTTOM,
     1                 USWIDTH,DSWIDTH,GRAV,IBCUS,NBDUS,IPR,EPS
 9070 FORMAT (A120, //,
     1 '>>> GENERAL DATA AND PARAMETERS <<<', /,
     2 '===================================',/,
     3 'Channel Length       =', F13.2, /,
     5 'dx                   =', F13.2, /,
     6 'Simulation Time      =', F13.2, /,
     7 'dt                   =', F13.2, /,
     8 'Manning Coefficient  =', E13.5, /,
     9 'Bed Slope            =', E13.5, /,
     % 'U/S Bottom Elevation =', F13.2, /,
     1 'U/S Bottom Width     =', F13.2, /,
     2 'D/S Bottom Width     =', F13.2, /,
     3 'Gravity              =', F13.2, /,
     4 'Type of U/S Boundary =', I10, /,
     5 '# of data on U/S B.C =', I10, /,
     6 'Output Frequency     =', I10, /,
     7 'Relative Accuracy    =', E13.5, /,
     8 '===================================',//)
```



```
      WRITE(NUMF,9080) (TAB(J),INODE(J),TAB(J),INODE(J), J=1,NPLOT)
 9080 FORMAT(
     % '>>> WATERDEPTHS AND DISCHARGES DURING SIMULATION <<<', /,
     % '============================================',
     % '===========================================', /,'  Time ',
     % 10(A1,'    H(',I2,')',A1,'    Q(',I2,')') )
      WRITE(NUMF,9090)
 9090 FORMAT(
     % '--------------------------------------------',
     % '-------------------------------------------')


  999 RETURN
      END

C0***6****1*********2*********3*********4*********5*********6*********77
      SUBROUTINE LISTER(NUMF,T,DEPTH,DISCH,NS,NPLOT,INODE,TAB)
C-----------------------------------------------------------------------
      INTEGER INODE(NPLOT)
      REAL DEPTH(NS),DISCH(NS)
      CHARACTER TAB(NS)*1

      WRITE(NUMF,9000) T,
     1     ( TAB(J),DEPTH(INODE(J)),TAB(J),DISCH(INODE(J)), J=1,NPLOT )
 9000 FORMAT(F7.1,10(A1,F9.3))

  999 RETURN
      END

C0***6****1*********2*********3*********4*********5*********6*********77
      SUBROUTINE TAILER(NUMF,NS,DX,H0,Q0,DEPTH,DISCH,TMAX,GRAV,V1,V2,
     1                  UMIN,UMAX,IMIN,HMIN,IMAX,HMAX,CRMIN,CRMAX,TAB)
C-----------------------------------------------------------------------
      REAL H0(NS),DEPTH(NS),DISCH(NS)
      CHARACTER TAB(NS)*1

      WRITE(NUMF,9080) TMAX
 9080 FORMAT(
     % '============================================',
     % '===========================================', ///,
     % ' WATER PROFILE AND DISCHARGES ALONG THE CHANNEL',/,
     % '================================================',/,
     % ' Channel      t = 0.0 sec      t =', F8.1,' sec'/,
     % '  Reach     Depth     Disch     Depth     Disch',/,
     % '------------------------------------------------')

      DO 10 I=1,NS
 10   WRITE(NUMF,9090) (I-1)*DX,TAB(1),H0(I),TAB(1),Q0,
     1                 TAB(1),DEPTH(I),TAB(1),DISCH(I)
 9090 FORMAT(F7.1,4(A1,F9.3))

      XMIN = (IMIN-1)*DX
      XMAX = (IMAX-1)*DX
      CMIN = WAVESPEED(XMIN,HMIN,GRAV)
      CMAX = WAVESPEED(XMAX,HMAX,GRAV)
      VP   = (V2-V1)/V1*100.0
```

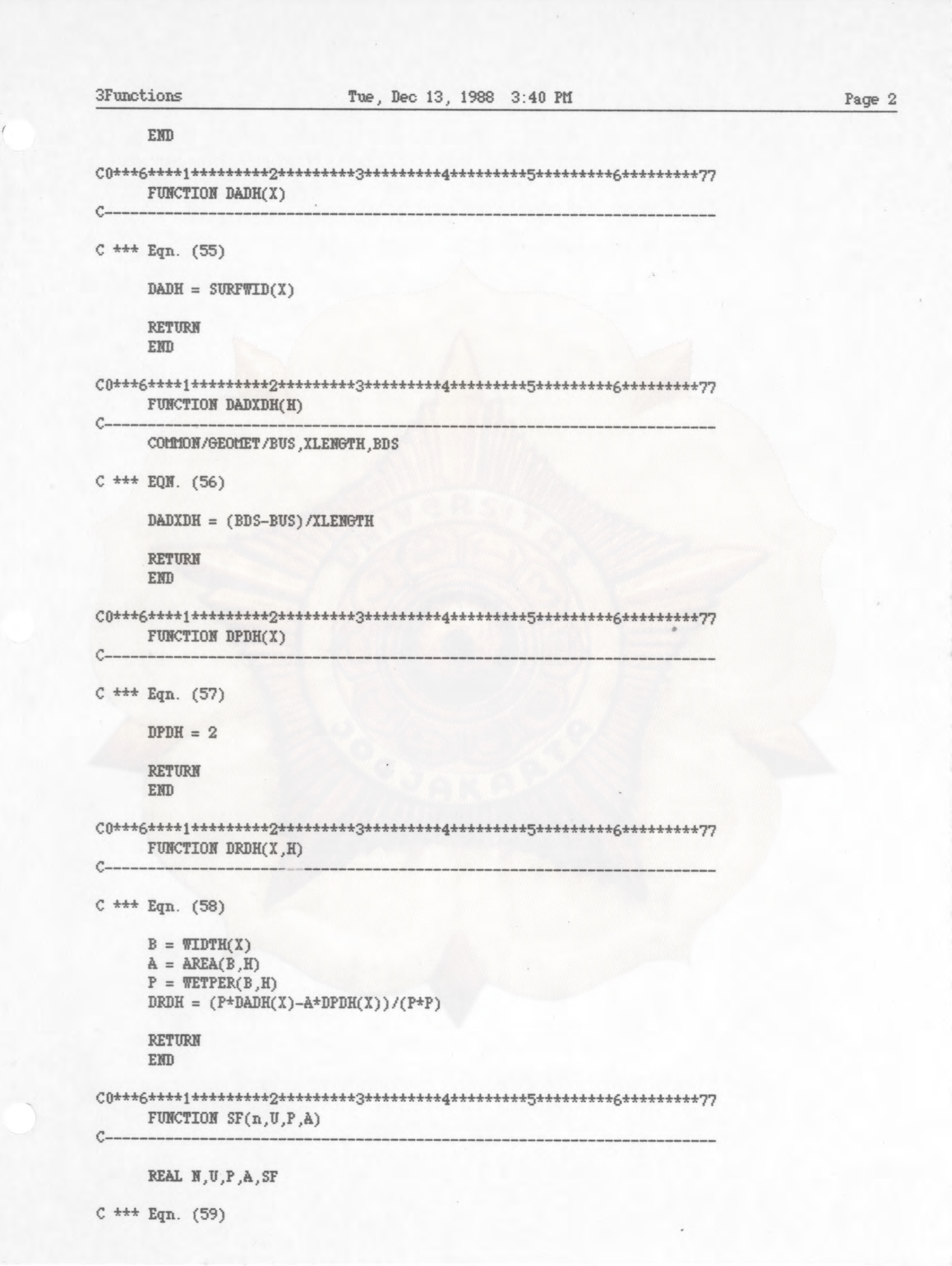


```
      WRITE(NUMF,*) '==============================================='

      WRITE(*,9100) UMIN,UMAX,HMIN,HMAX,CMIN,CMAX,CRMIN,CRMAX,V1,V2,VP
      WRITE(NUMF,9100) UMIN,UMAX,HMIN,HMAX,
     1                 CMIN,CMAX,CRMIN,CRMAX,V1,V2,VP
 9100 FORMAT(//,
     % 'STATISTICS OF THE SIMULATION',/,
     % '===========================',/,
     % 'Min. Velocity   =',F11.3,/,
     % 'Max. Velocity   =',F11.3,/,
     % 'Min. Waterdepth =',F11.3,/,
     % 'Max. Waterdepth =',F11.3,/,
     % 'Min. Celerity   =',F11.3,/,
     % 'Max. Celerity   =',F11.3,/,
     % '---------------------------',/,
     % 'Min. Courant #  =',F11.3,/,
     % 'Max. Courant #  =',F11.3,/,
     % '---------------------------',/,
     % 'Volume Inflow   =',E11.4,/,
     % 'Volume Outflow  =',E11.4,/,
     % 'Percentage      =',F9.3,' %',/,
     % '===========================')

  999 RETURN
      END
```

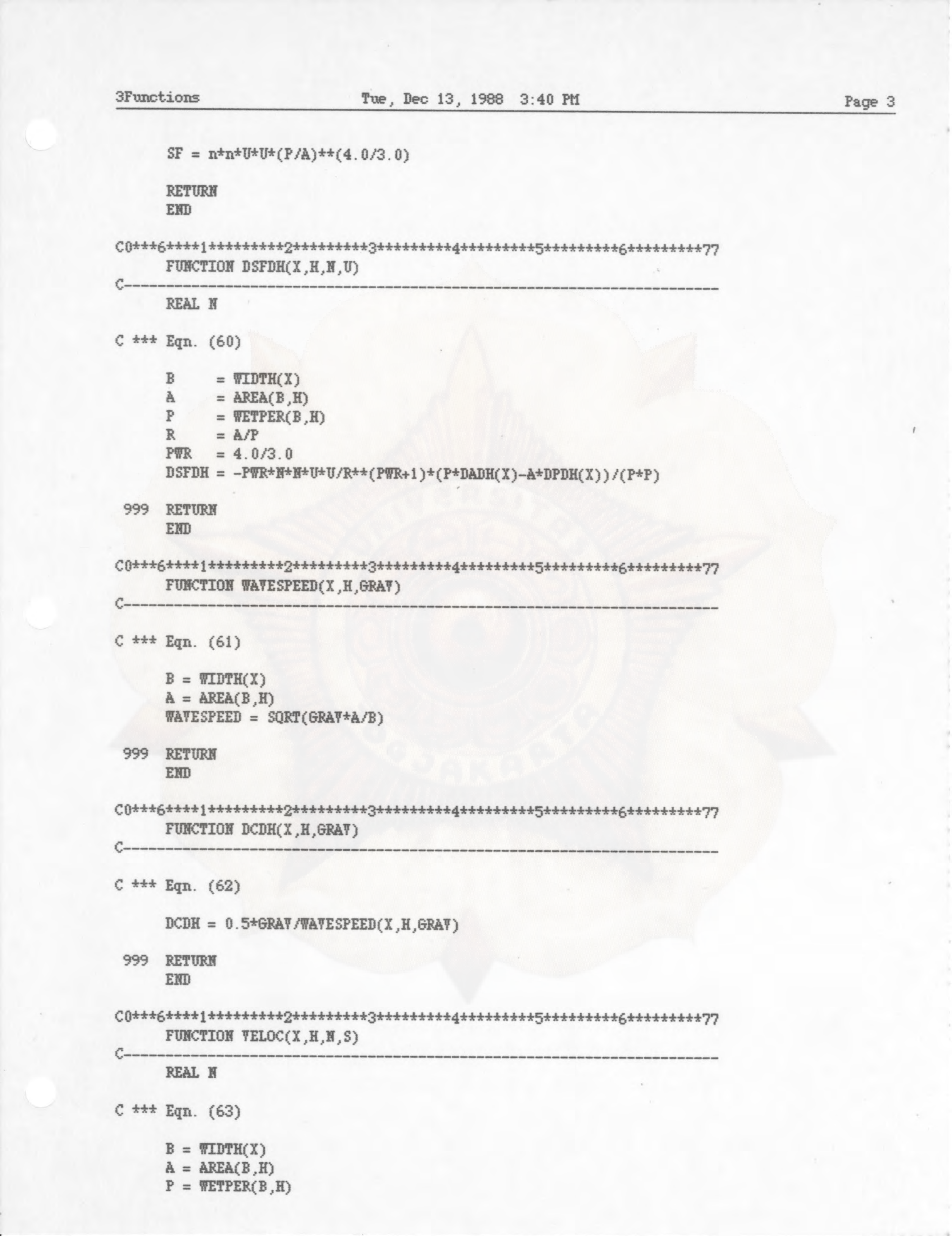


3Functions Tue, Dec 13, 1988 3:40 PM Page 1

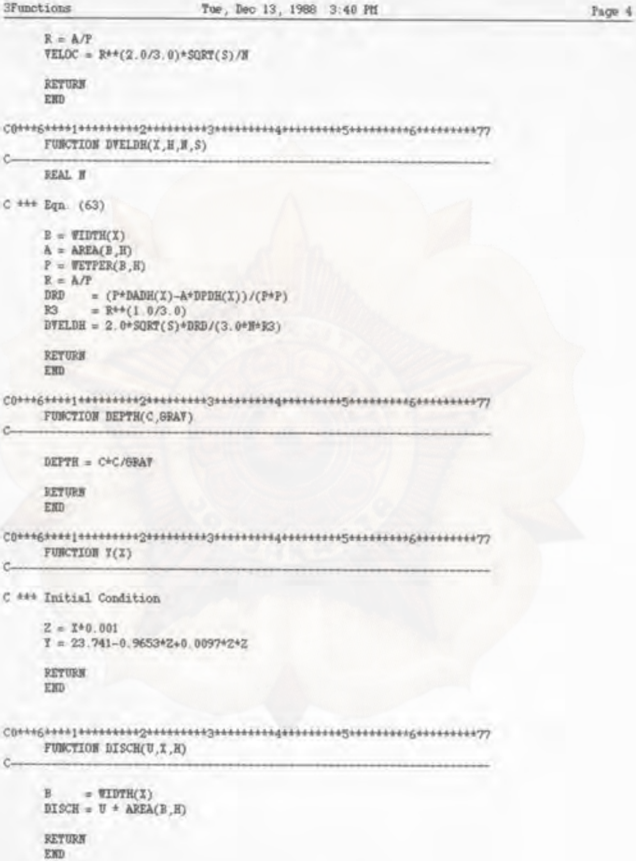
```
C0***6****1*********2*********3*********4*********5*********6*********77
      FUNCTION WIDTH(X)
C-----------------------------------------------------------------------
      COMMON/GEOMET/BUS,XLENGTH,BDS

C *** Eqn. (50)

      WIDTH = BUS + X*(BDS-BUS)/XLENGTH

      RETURN
      END

C0***6****1*********2*********3*********4*********5*********6*********77
      FUNCTION AREA(B,H)
C-----------------------------------------------------------------------

C *** Eqn. (51)

      AREA = B*H

      RETURN
      END

C0***6****1*********2*********3*********4*********5*********6*********77
      FUNCTION WETPER(B,H)
C-----------------------------------------------------------------------

C *** Eqn. (52)

      WETPER = B+2.0*H

      RETURN
      END

C0***6****1*********2*********3*********4*********5*********6*********77
      FUNCTION SURFWID(X)
C-----------------------------------------------------------------------
      COMMON/GEOMET/BUS,XLENGTH,BDS

C *** Eqn. (53)

      SURFWID = WIDTH(X)

      RETURN
      END

C0***6****1*********2*********3*********4*********5*********6*********77
      FUNCTION DADX(H)
C-----------------------------------------------------------------------
      COMMON/GEOMET/BUS,XLENGTH,BDS

C *** EQN. (54)

      DADX = H*(BDS-BUS)/XLENGTH

      RETURN
```



```
      END

C0***6****1*********2*********3*********4*********5*********6*********77
      FUNCTION DADH(X)
C-----------------------------------------------------------------------

C *** Eqn. (55)

      DADH = SURFWID(X)

      RETURN
      END

C0***6****1*********2*********3*********4*********5*********6*********77
      FUNCTION DADXDH(H)
C-----------------------------------------------------------------------
      COMMON/GEOMET/BUS,XLENGTH,BDS

C *** EQN. (56)

      DADXDH = (BDS-BUS)/XLENGTH

      RETURN
      END

C0***6****1*********2*********3*********4*********5*********6*********77
      FUNCTION DPDH(X)
C-----------------------------------------------------------------------

C *** Eqn. (57)

      DPDH = 2

      RETURN
      END

C0***6****1*********2*********3*********4*********5*********6*********77
      FUNCTION DRDH(X,H)
C-----------------------------------------------------------------------

C *** Eqn. (58)

      B = WIDTH(X)
      A = AREA(B,H)
      P = WETPER(B,H)
      DRDH = (P*DADH(X)-A*DPDH(X))/(P*P)

      RETURN
      END

C0***6****1*********2*********3*********4*********5*********6*********77
      FUNCTION SF(n,U,P,A)
C-----------------------------------------------------------------------

      REAL N,U,P,A,SF

C *** Eqn. (59)
```

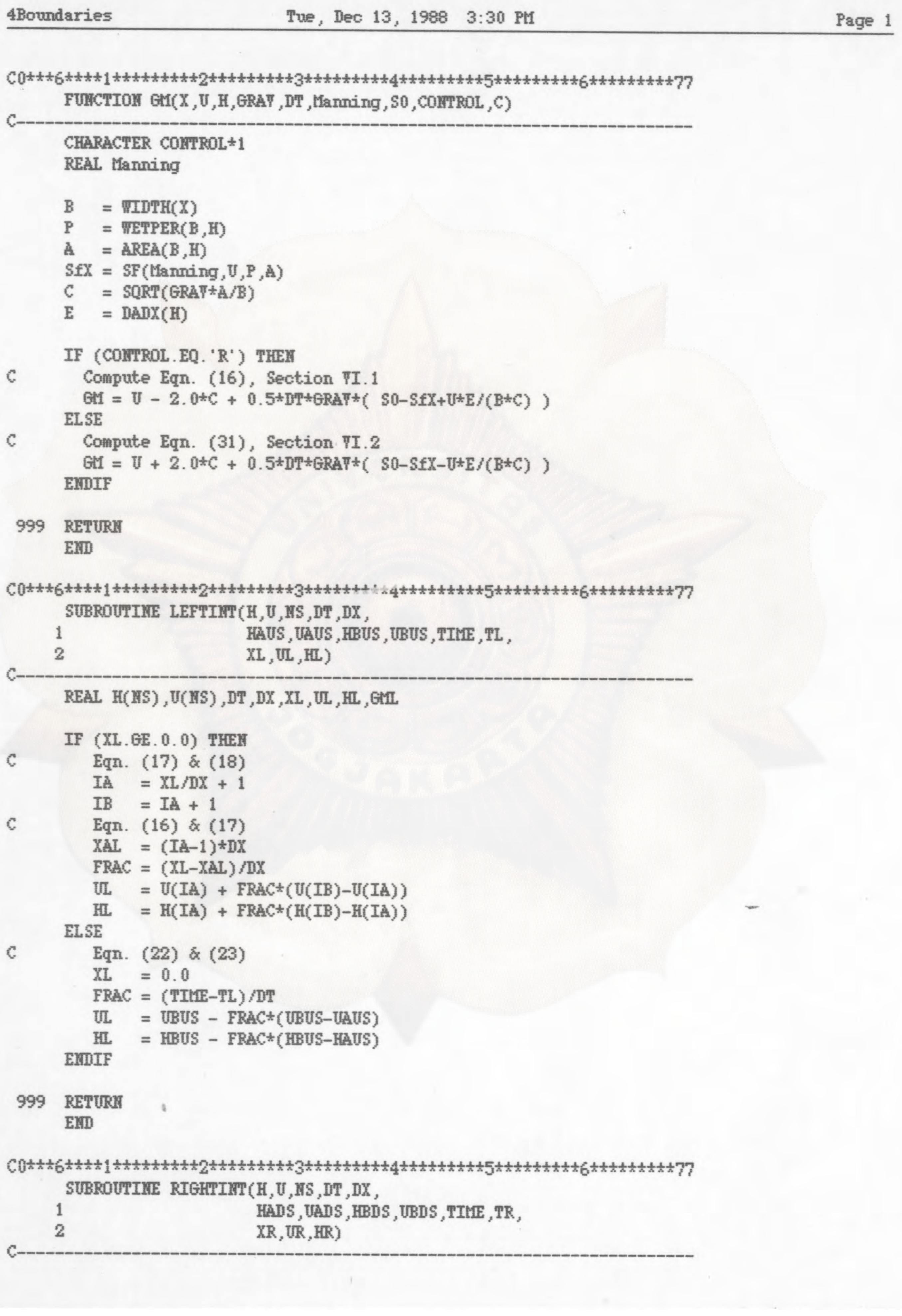


```
      SF = n*n*U*U*(P/A)**(4.0/3.0)

      RETURN
      END

C0***6****1*********2*********3*********4*********5*********6*********77
      FUNCTION DSFDH(X,H,N,U)
C-----------------------------------------------------------------------
      REAL N

C *** Eqn. (60)

      B     = WIDTH(X)
      A     = AREA(B,H)
      P     = WETPER(B,H)
      R     = A/P
      PWR   = 4.0/3.0
      DSFDH = -PWR*N*N*U*U/R**(PWR+1)*(P*DADH(X)-A*DPDH(X))/(P*P)

 999  RETURN
      END

C0***6****1*********2*********3*********4*********5*********6*********77
      FUNCTION WAVESPEED(X,H,GRAV)
C-----------------------------------------------------------------------

C *** Eqn. (61)

      B = WIDTH(X)
      A = AREA(B,H)
      WAVESPEED = SQRT(GRAV*A/B)

 999  RETURN
      END

C0***6****1*********2*********3*********4*********5*********6*********77
      FUNCTION DCDH(X,H,GRAV)
C-----------------------------------------------------------------------

C *** Eqn. (62)

      DCDH = 0.5*GRAV/WAVESPEED(X,H,GRAV)

 999  RETURN
      END

C0***6****1*********2*********3*********4*********5*********6*********77
      FUNCTION VELOC(X,H,N,S)
C-----------------------------------------------------------------------
      REAL N

C *** Eqn. (63)

      B = WIDTH(X)
      A = AREA(B,H)
      P = WETPER(B,H)
```

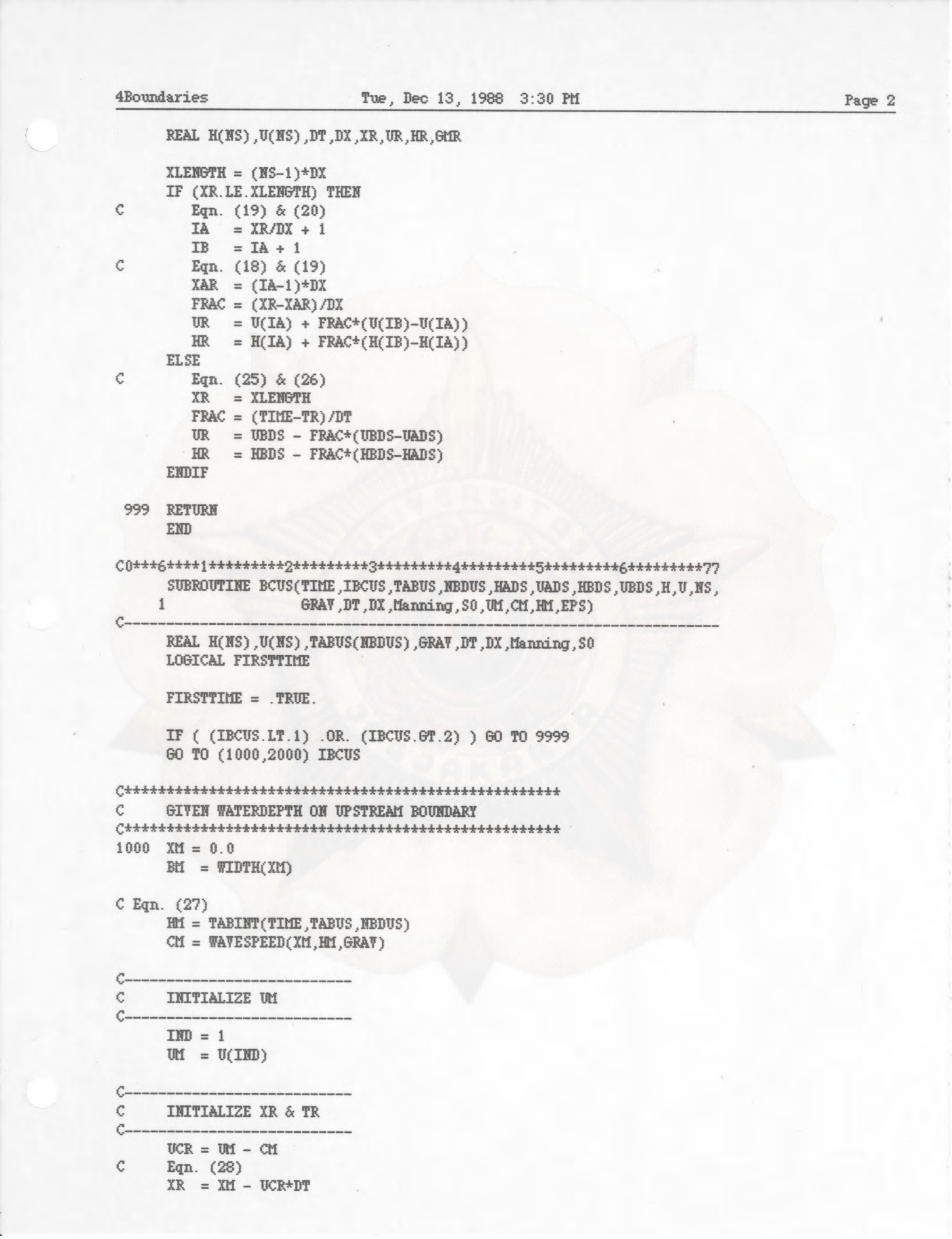


3Functions	Tue, Dec 13, 1988 3:40 PM	Page 4

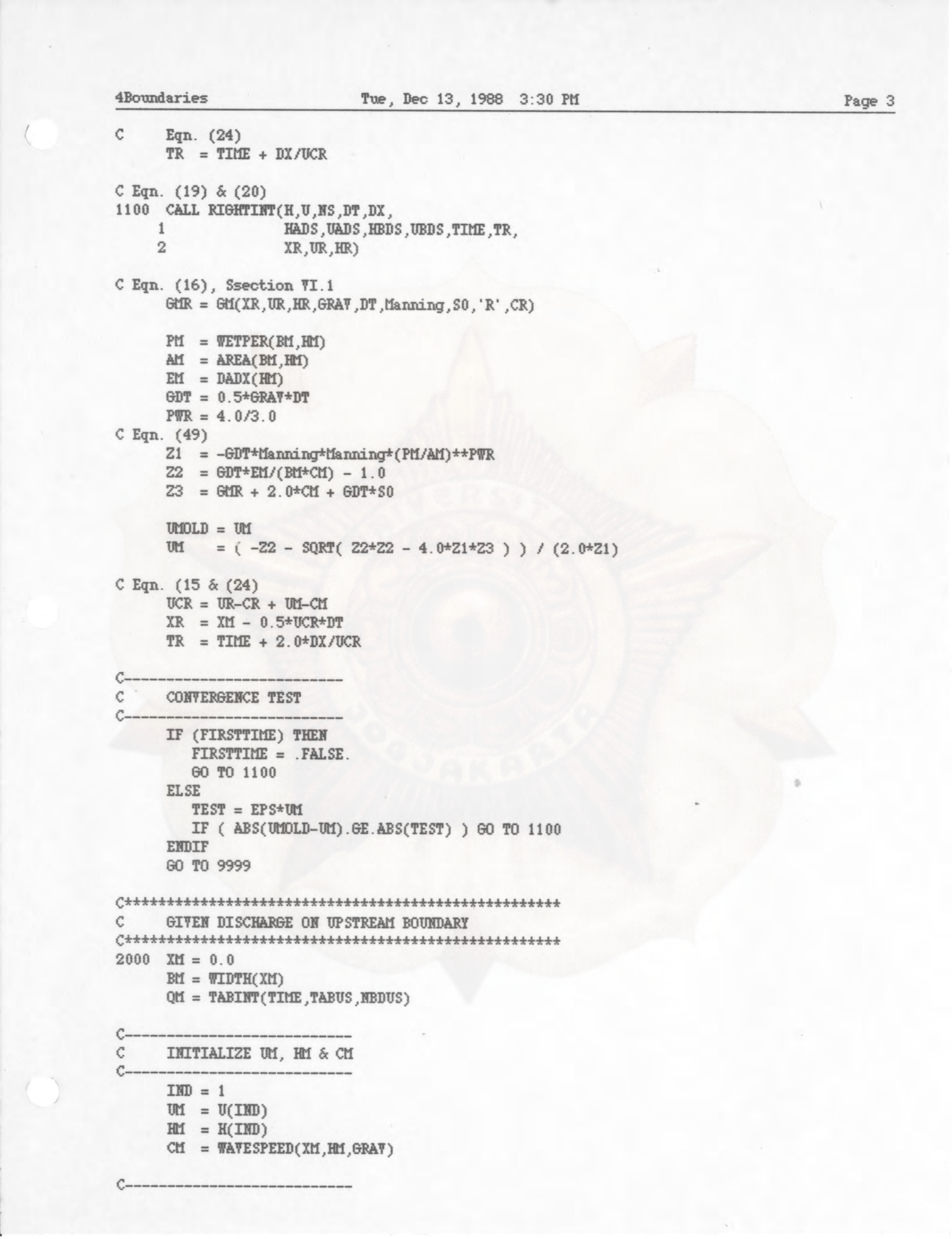
```
      R = A/P
      VELOC = R**(2.0/3.0)*SQRT(S)/N

      RETURN
      END

C0***6****1*********2*********3*********4*********5*********6*********77
      FUNCTION DVELDH(X,H,N,S)
C-----------------------------------------------------------------------
      REAL N

C *** Eqn. (63)

      B = WIDTH(X)
      A = AREA(B,H)
      P = WETPER(B,H)
      R = A/P
      DRD    = (P*DADH(X)-A*DPDH(X))/(P*P)
      R3     = R**(1.0/3.0)
      DVELDH = 2.0*SQRT(S)*DRD/(3.0*N*R3)

      RETURN
      END

C0***6****1*********2*********3*********4*********5*********6*********77
      FUNCTION DEPTH(C,GRAV)
C-----------------------------------------------------------------------

      DEPTH = C*C/GRAV

      RETURN
      END

C0***6****1*********2*********3*********4*********5*********6*********77
      FUNCTION Y(X)
C-----------------------------------------------------------------------

C *** Initial Condition

      Z = X*0.001
      Y = 23.741-0.9653*Z+0.0097*Z*Z

      RETURN
      END

C0***6****1*********2*********3*********4*********5*********6*********77
      FUNCTION DISCH(U,X,H)
C-----------------------------------------------------------------------

      B     = WIDTH(X)
      DISCH = U * AREA(B,H)

      RETURN
      END
```



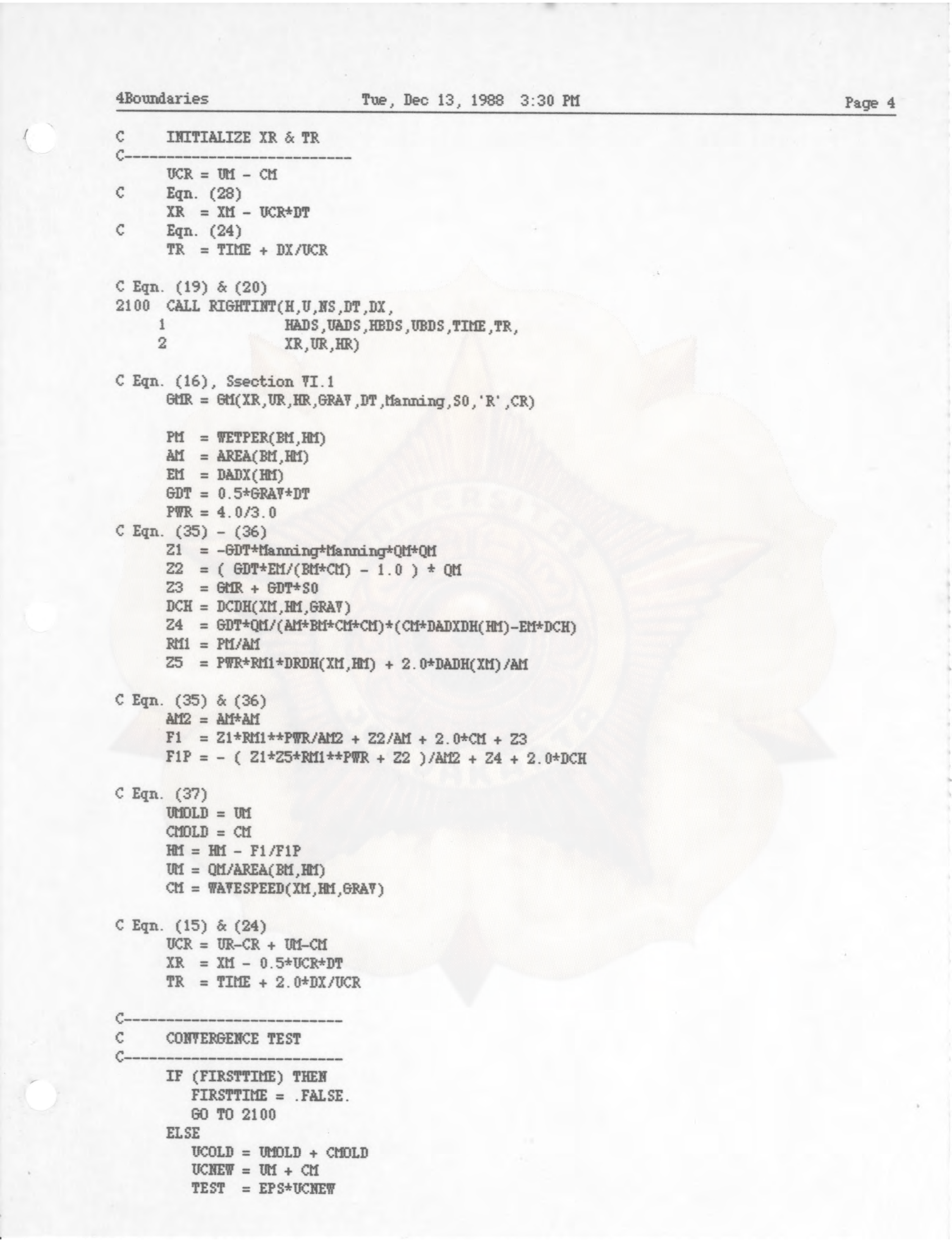
```
C0***6****1*********2*********3*********4*********5*********6*********77
      FUNCTION GM(X,U,H,GRAV,DT,Manning,S0,CONTROL,C)
C-----------------------------------------------------------------------
      CHARACTER CONTROL*1
      REAL Manning

      B   = WIDTH(X)
      P   = WETPER(B,H)
      A   = AREA(B,H)
      SfX = SF(Manning,U,P,A)
      C   = SQRT(GRAV*A/B)
      E   = DADX(H)

      IF (CONTROL.EQ.'R') THEN
C       Compute Eqn. (16), Section VI.1
        GM = U - 2.0*C + 0.5*DT*GRAV*( S0-SfX+U*E/(B*C) )
      ELSE
C       Compute Eqn. (31), Section VI.2
        GM = U + 2.0*C + 0.5*DT*GRAV*( S0-SfX-U*E/(B*C) )
      ENDIF

 999  RETURN
      END

C0***6****1*********2*********3*********4*********5*********6*********77
      SUBROUTINE LEFTINT(H,U,NS,DT,DX,
     1                   HAUS,UAUS,HBUS,UBUS,TIME,TL,
     2                   XL,UL,HL)
C-----------------------------------------------------------------------
      REAL H(NS),U(NS),DT,DX,XL,UL,HL,GML

      IF (XL.GE.0.0) THEN
C         Eqn. (17) & (18)
          IA   = XL/DX + 1
          IB   = IA + 1
C         Eqn. (16) & (17)
          XAL  = (IA-1)*DX
          FRAC = (XL-XAL)/DX
          UL   = U(IA) + FRAC*(U(IB)-U(IA))
          HL   = H(IA) + FRAC*(H(IB)-H(IA))
      ELSE
C         Eqn. (22) & (23)
          XL   = 0.0
          FRAC = (TIME-TL)/DT
          UL   = UBUS - FRAC*(UBUS-UAUS)
          HL   = HBUS - FRAC*(HBUS-HAUS)
      ENDIF

 999  RETURN
      END

C0***6****1*********2*********3*********4*********5*********6*********77
      SUBROUTINE RIGHTINT(H,U,NS,DT,DX,
     1                    HADS,UADS,HBDS,UBDS,TIME,TR,
     2                    XR,UR,HR)
C-----------------------------------------------------------------------
```



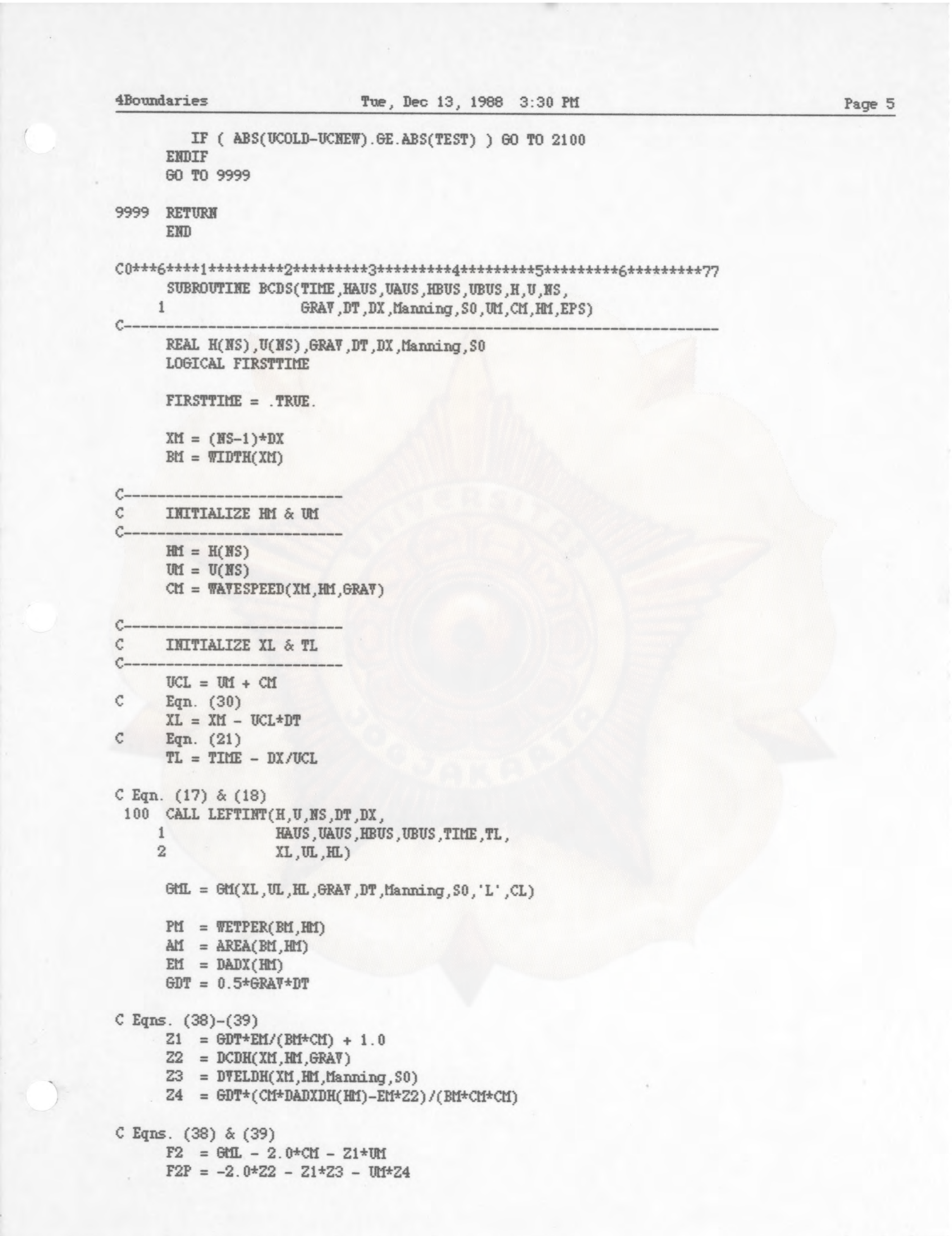
```
      REAL H(NS),U(NS),DT,DX,XR,UR,HR,GMR

      XLENGTH = (NS-1)*DX
      IF (XR.LE.XLENGTH) THEN
C        Eqn. (19) & (20)
         IA   = XR/DX + 1
         IB   = IA + 1
C        Eqn. (18) & (19)
         XAR  = (IA-1)*DX
         FRAC = (XR-XAR)/DX
         UR   = U(IA) + FRAC*(U(IB)-U(IA))
         HR   = H(IA) + FRAC*(H(IB)-H(IA))
      ELSE
C        Eqn. (25) & (26)
         XR   = XLENGTH
         FRAC = (TIME-TR)/DT
         UR   = UBDS - FRAC*(UBDS-UADS)
         HR   = HBDS - FRAC*(HBDS-HADS)
      ENDIF

 999  RETURN
      END

C0***6*****1*********2*********3*********4*********5*********6*********77
      SUBROUTINE BCUS(TIME,IBCUS,TABUS,NBDUS,HADS,UADS,HBDS,UBDS,H,U,NS,
     1                GRAV,DT,DX,Manning,S0,UM,CM,HM,EPS)
C-----------------------------------------------------------------------
      REAL H(NS),U(NS),TABUS(NBDUS),GRAV,DT,DX,Manning,S0
      LOGICAL FIRSTTIME

      FIRSTTIME = .TRUE.

      IF ( (IBCUS.LT.1) .OR. (IBCUS.GT.2) ) GO TO 9999
      GO TO (1000,2000) IBCUS

C****************************************************
C     GIVEN WATERDEPTH ON UPSTREAM BOUNDARY
C****************************************************
1000  XM = 0.0
      BM  = WIDTH(XM)

C Eqn. (27)
      HM = TABINT(TIME,TABUS,NBDUS)
      CM = WAVESPEED(XM,HM,GRAV)

C---------------------------
C     INITIALIZE UM
C---------------------------
      IND = 1
      UM  = U(IND)

C---------------------------
C     INITIALIZE XR & TR
C---------------------------
      UCR = UM - CM
C     Eqn. (28)
      XR  = XM - UCR*DT
```



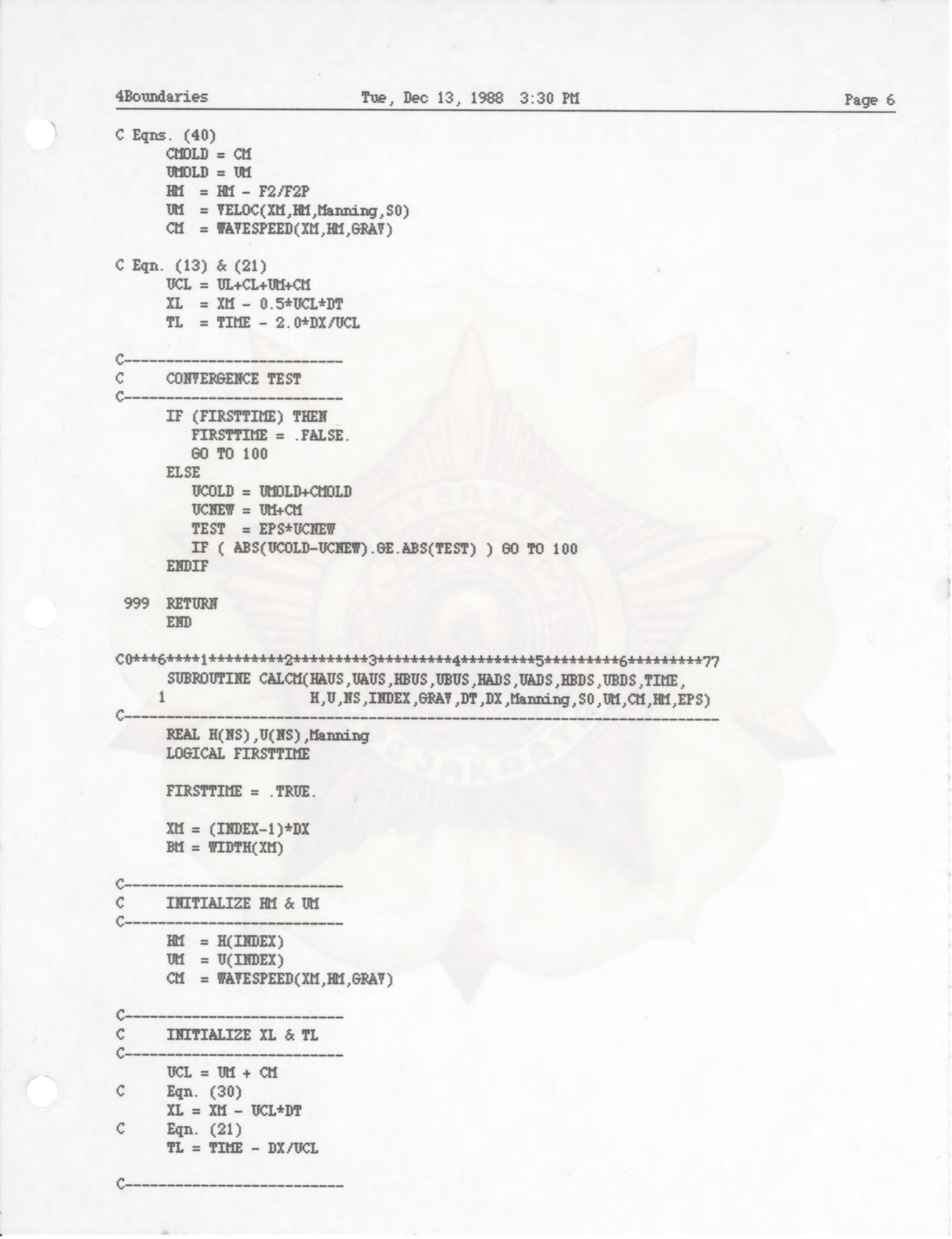
```
C     Eqn. (24)
      TR  = TIME + DX/UCR

C Eqn. (19) & (20)
1100  CALL RIGHTINT(H,U,NS,DT,DX,
     1              HADS,UADS,HBDS,UBDS,TIME,TR,
     2              XR,UR,HR)

C Eqn. (16), Ssection VI.1
      GMR = GM(XR,UR,HR,GRAV,DT,Manning,S0,'R',CR)

      PM  = WETPER(BM,HM)
      AM  = AREA(BM,HM)
      EM  = DADX(HM)
      GDT = 0.5*GRAV*DT
      PWR = 4.0/3.0
C Eqn. (49)
      Z1  = -GDT*Manning*Manning*(PM/AM)**PWR
      Z2  = GDT*EM/(BM*CM) - 1.0
      Z3  = GMR + 2.0*CM + GDT*S0

      UMOLD = UM
      UM    = ( -Z2 - SQRT( Z2*Z2 - 4.0*Z1*Z3 ) ) / (2.0*Z1)

C Eqn. (15 & (24)
      UCR = UR-CR + UM-CM
      XR  = XM - 0.5*UCR*DT
      TR  = TIME + 2.0*DX/UCR

C-------------------------
C     CONVERGENCE TEST
C-------------------------
      IF (FIRSTTIME) THEN
         FIRSTTIME = .FALSE.
         GO TO 1100
      ELSE
         TEST = EPS*UM
         IF ( ABS(UMOLD-UM).GE.ABS(TEST) ) GO TO 1100
      ENDIF
      GO TO 9999

C*****************************************************
C     GIVEN DISCHARGE ON UPSTREAM BOUNDARY
C*****************************************************
2000  XM = 0.0
      BM = WIDTH(XM)
      QM = TABINT(TIME,TABUS,NBDUS)

C--------------------------
C     INITIALIZE UM, HM & CM
C--------------------------
      IND = 1
      UM  = U(IND)
      HM  = H(IND)
      CM  = WAVESPEED(XM,HM,GRAV)

C--------------------------
```



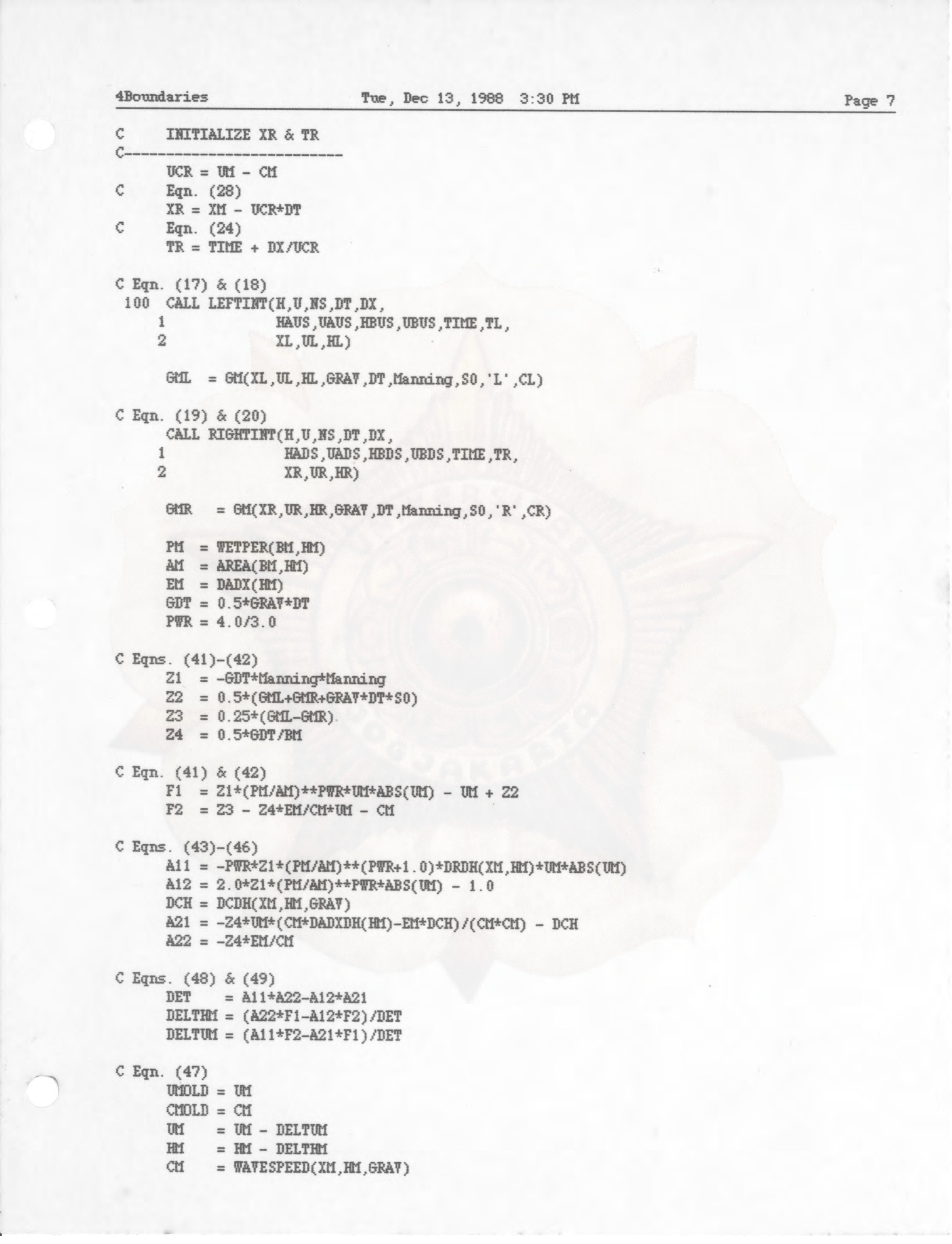
```
C     INITIALIZE XR & TR
C---------------------------
      UCR = UM - CM
C     Eqn. (28)
      XR  = XM - UCR*DT
C     Eqn. (24)
      TR  = TIME + DX/UCR

C Eqn. (19) & (20)
2100  CALL RIGHTINT(H,U,NS,DT,DX,
     1              HADS,UADS,HBDS,UBDS,TIME,TR,
     2              XR,UR,HR)

C Eqn. (16), Ssection VI.1
      GMR = GM(XR,UR,HR,GRAV,DT,Manning,S0,'R',CR)

      PM  = WETPER(BM,HM)
      AM  = AREA(BM,HM)
      EM  = DADX(HM)
      GDT = 0.5*GRAV*DT
      PWR = 4.0/3.0
C Eqn. (35) - (36)
      Z1  = -GDT*Manning*Manning*QM*QM
      Z2  = ( GDT*EM/(BM*CM) - 1.0 ) * QM
      Z3  = GMR + GDT*S0
      DCH = DCDH(XM,HM,GRAV)
      Z4  = GDT*QM/(AM*BM*CM*CM)*(CM*DADXDH(HM)-EM*DCH)
      RM1 = PM/AM
      Z5  = PWR*RM1*DRDH(XM,HM) + 2.0*DADH(XM)/AM

C Eqn. (35) & (36)
      AM2 = AM*AM
      F1  = Z1*RM1**PWR/AM2 + Z2/AM + 2.0*CM + Z3
      F1P = - ( Z1*Z5*RM1**PWR + Z2 )/AM2 + Z4 + 2.0*DCH

C Eqn. (37)
      UMOLD = UM
      CMOLD = CM
      HM = HM - F1/F1P
      UM = QM/AREA(BM,HM)
      CM = WAVESPEED(XM,HM,GRAV)

C Eqn. (15) & (24)
      UCR = UR-CR + UM-CM
      XR  = XM - 0.5*UCR*DT
      TR  = TIME + 2.0*DX/UCR

C--------------------------
C     CONVERGENCE TEST
C--------------------------
      IF (FIRSTTIME) THEN
         FIRSTTIME = .FALSE.
         GO TO 2100
      ELSE
         UCOLD = UMOLD + CMOLD
         UCNEW = UM + CM
         TEST  = EPS*UCNEW
```



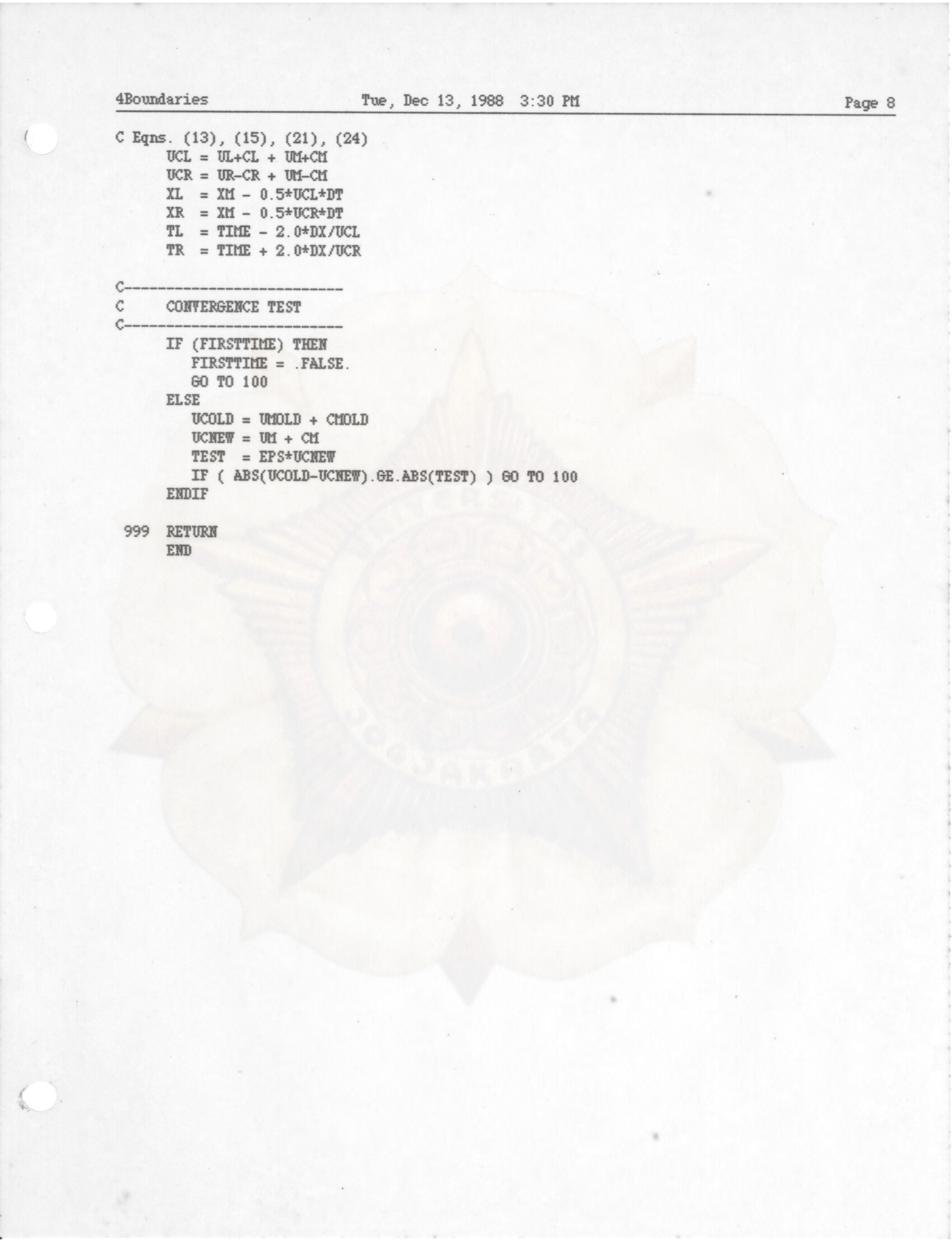
```
            IF ( ABS(UCOLD-UCNEW).GE.ABS(TEST) ) GO TO 2100
         ENDIF
         GO TO 9999

9999     RETURN
         END

C0***6****1*********2*********3*********4*********5*********6*********77
      SUBROUTINE BCDS(TIME,HAUS,UAUS,HBUS,UBUS,H,U,NS,
     1                GRAV,DT,DX,Manning,S0,UM,CM,HM,EPS)
C-----------------------------------------------------------------------
      REAL H(NS),U(NS),GRAV,DT,DX,Manning,S0
      LOGICAL FIRSTTIME

      FIRSTTIME = .TRUE.

      XM = (NS-1)*DX
      BM = WIDTH(XM)

C-------------------------
C     INITIALIZE HM & UM
C-------------------------
      HM = H(NS)
      UM = U(NS)
      CM = WAVESPEED(XM,HM,GRAV)

C-------------------------
C     INITIALIZE XL & TL
C-------------------------
      UCL = UM + CM
C     Eqn. (30)
      XL = XM - UCL*DT
C     Eqn. (21)
      TL = TIME - DX/UCL

C Eqn. (17) & (18)
 100  CALL LEFTINT(H,U,NS,DT,DX,
     1             HAUS,UAUS,HBUS,UBUS,TIME,TL,
     2             XL,UL,HL)

      GML = GM(XL,UL,HL,GRAV,DT,Manning,S0,'L',CL)

      PM  = WETPER(BM,HM)
      AM  = AREA(BM,HM)
      EM  = DADX(HM)
      GDT = 0.5*GRAV*DT

C Eqns. (38)-(39)
      Z1  = GDT*EM/(BM*CM) + 1.0
      Z2  = DCDH(XM,HM,GRAV)
      Z3  = DVELDH(XM,HM,Manning,S0)
      Z4  = GDT*(CM*DADXDH(HM)-EM*Z2)/(BM*CM*CM)

C Eqns. (38) & (39)
      F2  = GML - 2.0*CM - Z1*UM
      F2P = -2.0*Z2 - Z1*Z3 - UM*Z4
```



```
C Eqns. (40)
      CMOLD = CM
      UMOLD = UM
      HM  = HM - F2/F2P
      UM  = VELOC(XM,HM,Manning,S0)
      CM  = WAVESPEED(XM,HM,GRAV)

C Eqn. (13) & (21)
      UCL = UL+CL+UM+CM
      XL  = XM - 0.5*UCL*DT
      TL  = TIME - 2.0*DX/UCL

C--------------------------
C     CONVERGENCE TEST
C--------------------------
      IF (FIRSTTIME) THEN
         FIRSTTIME = .FALSE.
         GO TO 100
      ELSE
         UCOLD = UMOLD+CMOLD
         UCNEW = UM+CM
         TEST  = EPS*UCNEW
         IF ( ABS(UCOLD-UCNEW).GE.ABS(TEST) ) GO TO 100
      ENDIF

 999  RETURN
      END

C0***6****1*********2*********3*********4*********5*********6*********77
      SUBROUTINE CALCM(HAUS,UAUS,HBUS,UBUS,HADS,UADS,HBDS,UBDS,TIME,
     1                 H,U,NS,INDEX,GRAV,DT,DX,Manning,S0,UM,CM,HM,EPS)
C-----------------------------------------------------------------------
      REAL H(NS),U(NS),Manning
      LOGICAL FIRSTTIME

      FIRSTTIME = .TRUE.

      XM = (INDEX-1)*DX
      BM = WIDTH(XM)

C--------------------------
C     INITIALIZE HM & UM
C--------------------------
      HM  = H(INDEX)
      UM  = U(INDEX)
      CM  = WAVESPEED(XM,HM,GRAV)

C--------------------------
C     INITIALIZE XL & TL
C--------------------------
      UCL = UM + CM
C     Eqn. (30)
      XL = XM - UCL*DT
C     Eqn. (21)
      TL = TIME - DX/UCL

C--------------------------
```



```
C     INITIALIZE XR & TR
C-------------------------
      UCR = UM - CM
C     Eqn. (28)
      XR = XM - UCR*DT
C     Eqn. (24)
      TR = TIME + DX/UCR

C Eqn. (17) & (18)
 100  CALL LEFTINT(H,U,NS,DT,DX,
     1             HAUS,UAUS,HBUS,UBUS,TIME,TL,
     2             XL,UL,HL)

      GML  = GM(XL,UL,HL,GRAV,DT,Manning,S0,'L',CL)

C Eqn. (19) & (20)
      CALL RIGHTINT(H,U,NS,DT,DX,
     1              HADS,UADS,HBDS,UBDS,TIME,TR,
     2              XR,UR,HR)

      GMR   = GM(XR,UR,HR,GRAV,DT,Manning,S0,'R',CR)

      PM  = WETPER(BM,HM)
      AM  = AREA(BM,HM)
      EM  = DADX(HM)
      GDT = 0.5*GRAV*DT
      PWR = 4.0/3.0

C Eqns. (41)-(42)
      Z1  = -GDT*Manning*Manning
      Z2  = 0.5*(GML+GMR+GRAV*DT*S0)
      Z3  = 0.25*(GML-GMR)
      Z4  = 0.5*GDT/BM

C Eqn. (41) & (42)
      F1  = Z1*(PM/AM)**PWR*UM*ABS(UM) - UM + Z2
      F2  = Z3 - Z4*EM/CM*UM - CM

C Eqns. (43)-(46)
      A11 = -PWR*Z1*(PM/AM)**(PWR+1.0)*DRDH(XM,HM)*UM*ABS(UM)
      A12 = 2.0*Z1*(PM/AM)**PWR*ABS(UM) - 1.0
      DCH = DCDH(XM,HM,GRAV)
      A21 = -Z4*UM*(CM*DADXDH(HM)-EM*DCH)/(CM*CM) - DCH
      A22 = -Z4*EM/CM

C Eqns. (48) & (49)
      DET    = A11*A22-A12*A21
      DELTHM = (A22*F1-A12*F2)/DET
      DELTUM = (A11*F2-A21*F1)/DET

C Eqn. (47)
      UMOLD = UM
      CMOLD = CM
      UM    = UM - DELTUM
      HM    = HM - DELTHM
      CM    = WAVESPEED(XM,HM,GRAV)
```



```
C Eqns. (13), (15), (21), (24)
      UCL = UL+CL + UM+CM
      UCR = UR-CR + UM-CM
      XL  = XM - 0.5*UCL*DT
      XR  = XM - 0.5*UCR*DT
      TL  = TIME - 2.0*DX/UCL
      TR  = TIME + 2.0*DX/UCR

C---------------------------
C     CONVERGENCE TEST
C---------------------------
      IF (FIRSTTIME) THEN
         FIRSTTIME = .FALSE.
         GO TO 100
      ELSE
         UCOLD = UMOLD + CMOLD
         UCNEW = UM + CM
         TEST  = EPS*UCNEW
         IF ( ABS(UCOLD-UCNEW).GE.ABS(TEST) ) GO TO 100
      ENDIF

 999  RETURN
      END
```

VARIABLE-GRID METHOD

M.B. Sharifi
Term Project, 14 December 1988