1990

A general-purpose, statically-typed, purely functional programming language with type inference and lazy evaluation

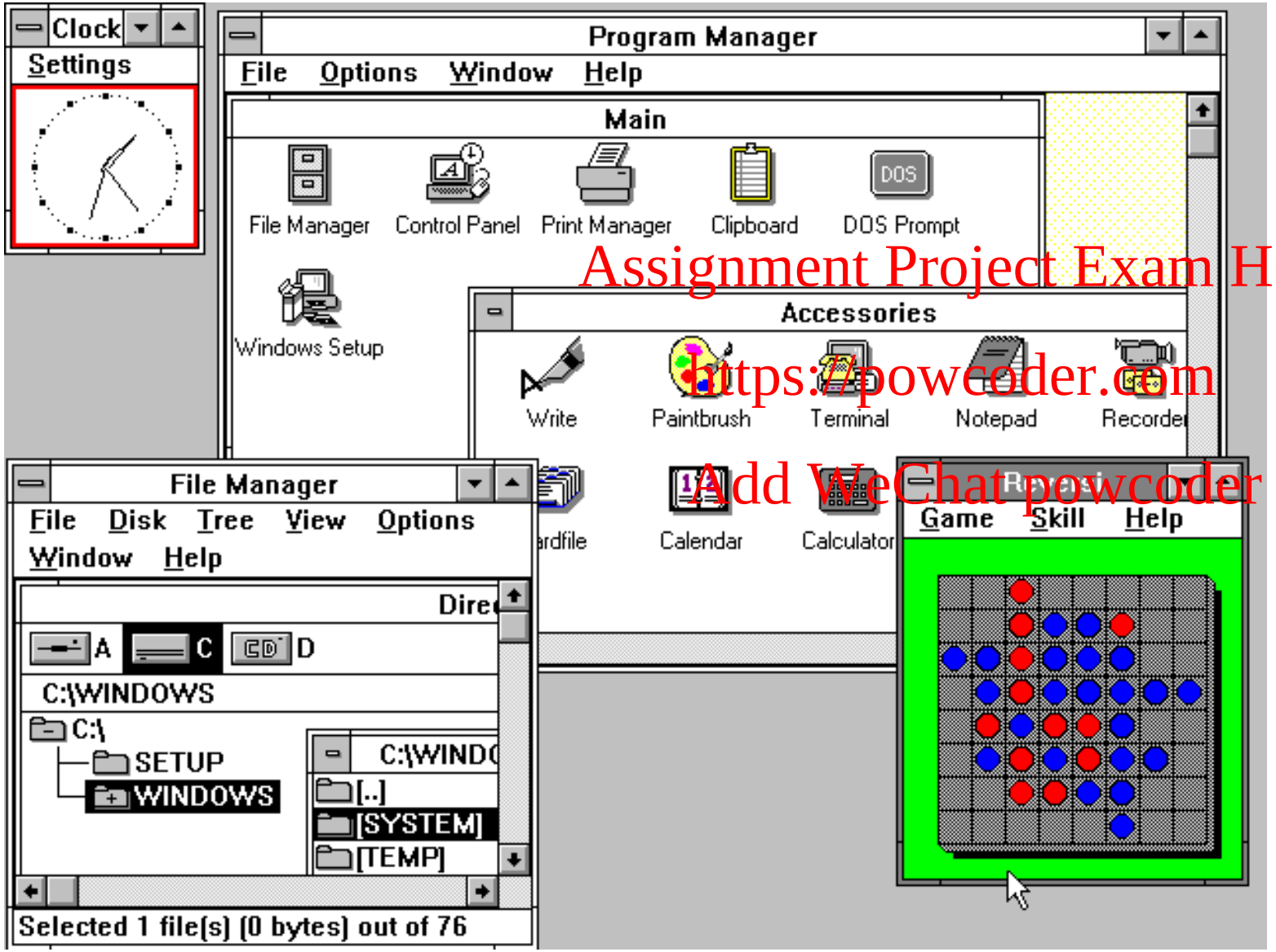
CMPT 383
Summer 2022
SFU Surrey

What was going on in 1990?

1990: First Appearances

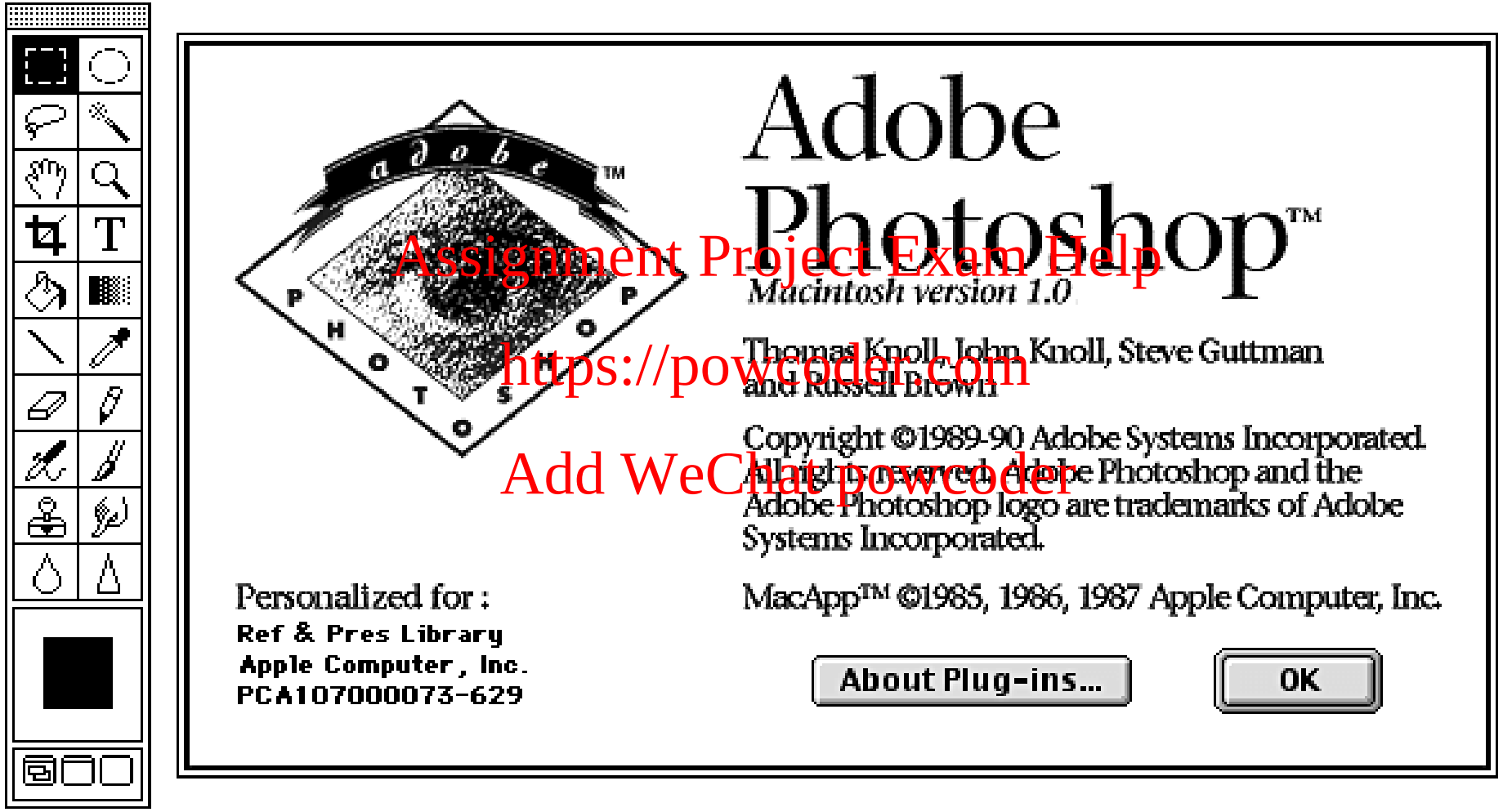
Hubble Space Telescope

1990: First Appearances



Windows 3.0

1990: First Appearances

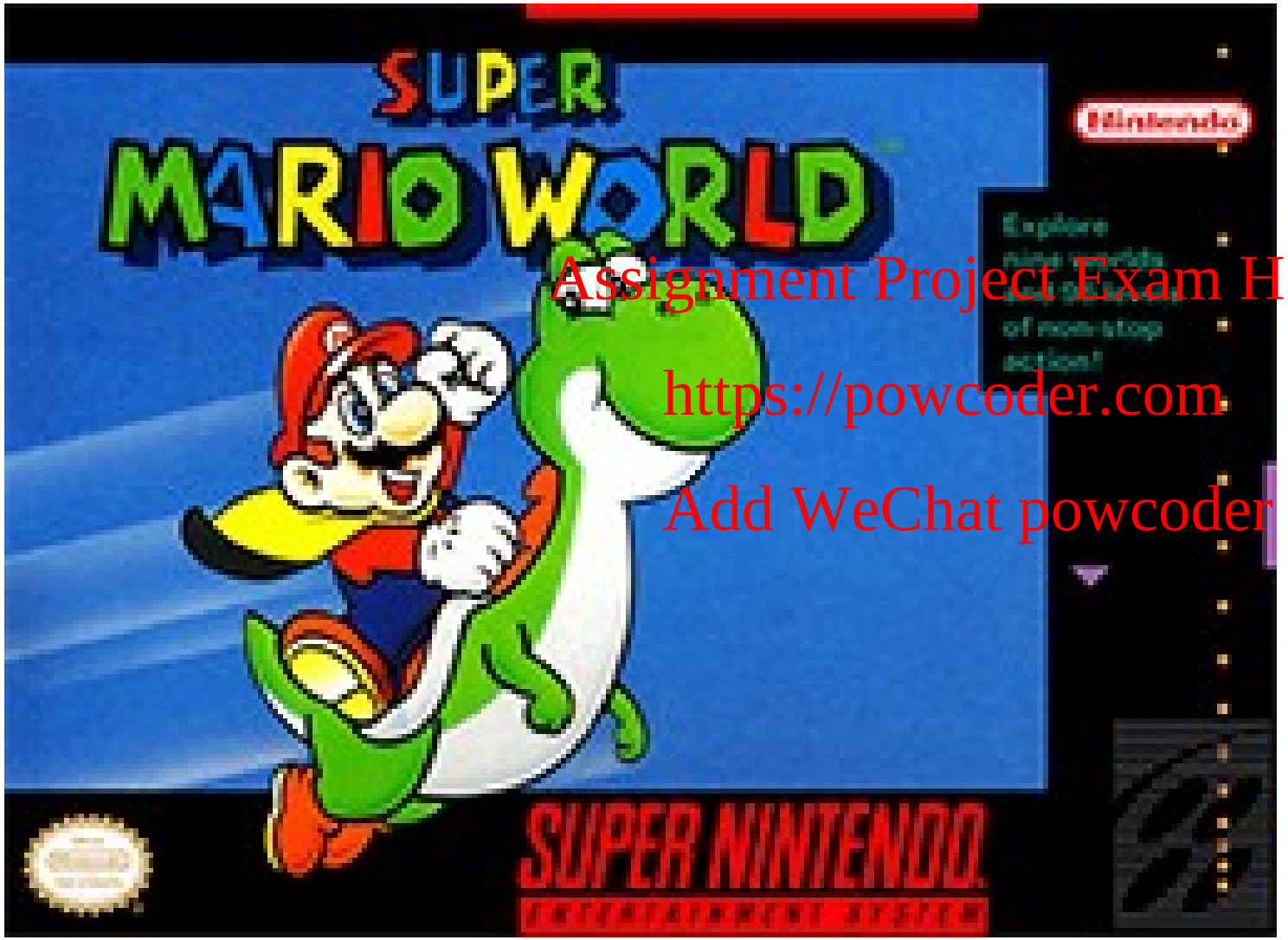


Photoshop 1.0

1990: Highest-grossing Arcade Game

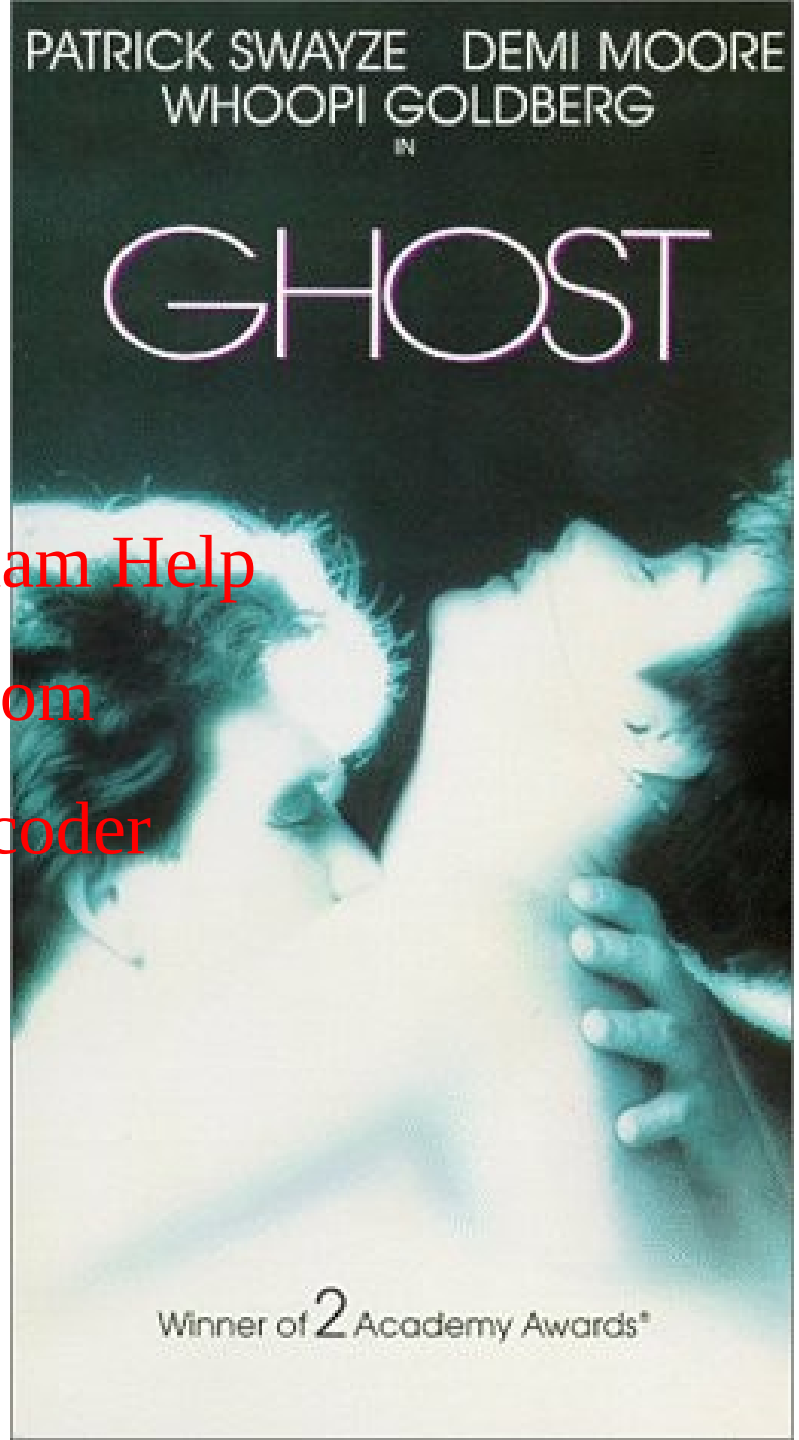
Teenage Mutant Ninja Turtles

1990: Highest-grossing Home Game



Super Mario World

1990: Highest-Grossing Movie



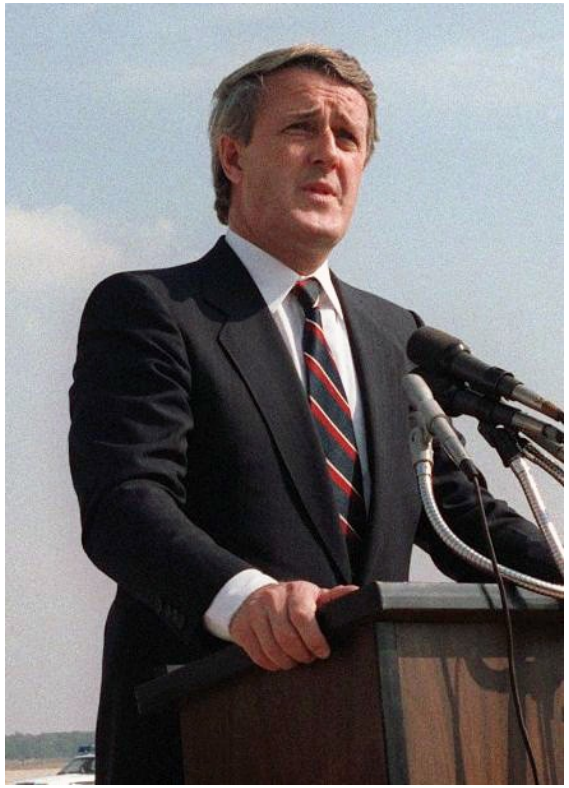
1990: Most Popular Song

Hold On

Wilson Philips

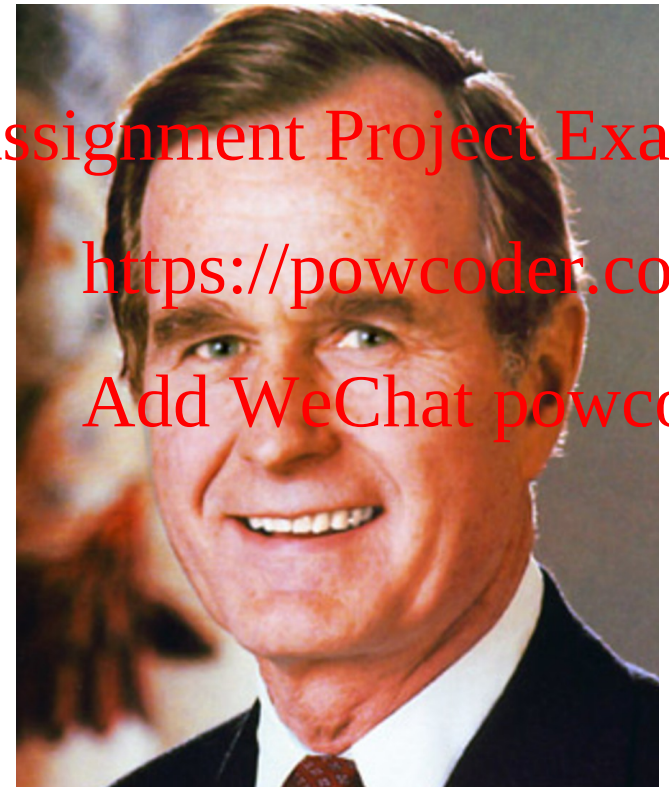
1990

Canadian PM



Brian Mulroney

US President



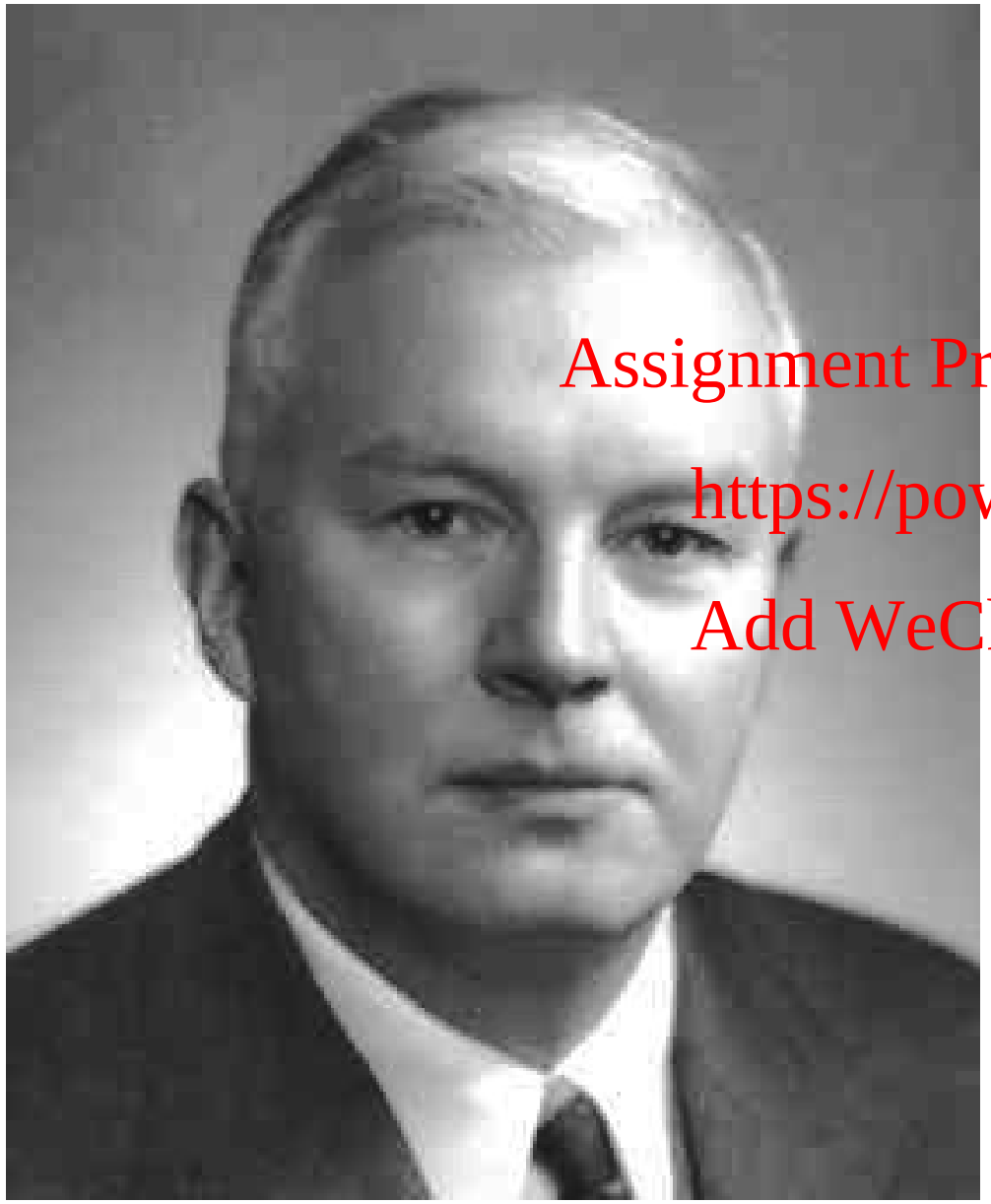
George H. W. Bush

This TV show started

1990

A general-purpose, statically-typed, purely functional programming language with type inference and lazy evaluation

Haskell's Namesake



Haskell Curry

1900 – 1982

Mathematician who did much of foundational work in the lambda calculus, the mathematical foundation of functional programming.

He wasn't involved with the Haskell language itself.

Haskell's Original Designers

1992 Haskell Working Group

Haskell Probably **Can't** Get You a Job!

Haskell has been most influential as a **research language** used by a subset of academic programming language researchers.

Similar to LISP, Haskell is filled with lots of ideas that have influenced many other languages.

What's Haskell Good for?

Programming language type systems.

Inspiration and new ways of thinking about programming.

Further development of the functional programming paradigm, aiming towards practicality.

What's Haskell Not so Good For?

- Haskell is often not as efficient as other languages.
  - Immutability can lead to inefficient solutions to problems that can be easily implemented efficiently in other languages.
- Compared to the most popular mainstream languages, Haskell is not used much in practice, so it can be hard to find support for it.
- Haskell's was born in academia and has a very mathematical flavor, which many programmers find quite different than C-style languages.
- Haskell syntax can be extremely terse, and can use many obscure operators.
- Haskell's strict typing can take some getting used to, and may be frustrating at first.

Measuring the performance of different programming languages is surprisingly difficult to do fairly. Check out

the Computer Language Benchmarks Game

.

Major Features of Haskell

Functions are first-class objects

- Functions can be passed and returned (closures)

Interpreted and Compiled

- Easy to evaluate individual expressions

Memory-safe and garbage-collected

- No manual memory deletion

Major Features of Scheme

Types

Statically typed, i.e. types checked **before** run-time

Type inference, i.e. often Haskell infer what types must be with the programmer needing to write them explicitly

Purely Functional

All values are **immutable**: once you create them, they cannot be changed

All functions are **pure**: no side-effects, output depends only on input

Lazy Evaluation

When you pass an expression to a Haskell function, the expression is passed **unevaluated** to the function

Major Features Not in Haskell

Features **not** part of Haskell

- No pointers
- No object-oriented programming
- No exceptions

Hello, World!

Follow the download instructions at https://www.haskell.org/downloads/

We'll mainly use **ghci**, the Glasgow Haskell interpreter. You can compile Haskell programs with **ghc**.


```
❯ ghci
GHCi, version 8.6.5: http://www.haskell.org/ghc/  :? for help
Prelude> putStrLn "Hello, World!"
Hello, World!
```